



ITS
Institut
Teknologi
Sepuluh Nopember

TUGAS AKHIR - KI141502

**PEMERINGKATAN PERGURUAN TINGGI
MENGUNAKAN METODE *HELMHOLTZ* UNTUK
MENGUKUR TINGKAT KESIAPTERAPAN TEKNOLOGI
PERGURUAN TINGGI DI INDONESIA**

Anugerah Yulindra Satyaji
NRP 5113 100 186

Dosen Pembimbing I
Prof. Drs. Ec. Ir. Rianarto Sarno, M.Sc.,Ph.D

Dosen Pembimbing II
Bagus Setya Rintyarna S.T. M.Kom.

JURUSAN TEKNIK INFORMATIKA
Fakultas Teknologi Informasi
Institut Teknologi Sepuluh Nopember
Surabaya 2017

TUGAS AKHIR - KI141502

**PEMERINGKATAN PERGURUAN TINGGI
MENGUNAKAN METODE *HEMLMHOLTZ* UNTUK
MENGUKUR TINGKAT KESIAPTERAPAN TEKNOLOGI
PERGURUAN TINGGI DI INDONESIA**

Anugerah Yulindra Satyaji
NRP 5113 100 186

Dosen Pembimbing I
Prof. Drs. Ec. Ir. Riyanarto Sarno, M.Sc.,Ph.D.

Dosen Pembimbing II
Bagus Setya Rintyarna S.T. M.Kom.

JURUSAN TEKNIK INFORMATIKA
Fakultas Teknologi Informasi
Institut Teknologi Sepuluh Nopember
Surabaya 2017

[Halaman ini sengaja dikosongkan]



FINAL PROJECT - KI141502

**UNIVERSITY RANKING USING *HELMHOLTZ* METHOD
TO MEASURE APPLICATION LEVEL TECHNOLOGY OF
UNIVERSITY IN INDONESIA**

**Anugerah Yulindra Satyaji
NRP 5113 100 186**

**Supervisor I
Prof. Drs. Ec. Ir. Riyanarto Sarno, M.Sc., Ph.D.**

**Supervisor II
Bagus Setya Rintyarna S.T. M.Kom.**

**INFORMATICS DEPARTMENT
Faculty of Information Technology
Institut Teknologi Sepuluh Nopember
Surabaya 2017**

[Halaman ini sengaja dikosongkan]

LEMBAR PENGESAHAN

PEMERINGKATAN PERGURUAN TINGGI MENGUNAKAN METODE *HELMHOLTZ* UNTUK MENGUKUR TINGKAT KESIAPTERAPAN TEKNOLOGI PERGURUAN TINGGI DI INDONESIA

TUGAS AKHIR

Diajukan Untuk Memenuhi Salah Satu Syarat
Memperoleh Gelar Sarjana Komputer
pada
Rumpun Mata Kuliah Manajemen Informasi
Program Studi S-1 Jurusan Teknik Informatika
Fakultas Teknologi Informasi
Institut Teknologi Sepuluh Nopember

Oleh

ANUGERAH YULINDRA SATYAJI

NRP. 5113 100 186

Disetujui oleh Dosen Pembimbing Tugas Akhir:

Prof. Drs. Ec. Ir. RIYANARTO SARMO

M.Sc., Ph.D.

NIP: 19590803 198601 1 001.

BAGUS SETYA RINTYARNA

M.Kom.

NIK: 0509502



**SURABAYA
JUNI, 2017**

[Halaman ini sengaja dikosongkan]

PEMERINGKATAN PERGURUAN TINGGI MENGUNAKAN METODE *HELMHOLTZ* UNTUK MENGUKUR TINGKAT KESIAPTERAPAN TEKNOLOGI PERGURUAN TINGGI DI INDONESIA

Nama	: Anugerah Yulindra Satyaji
NRP	: 5113100186
Jurusan	: Teknik Informatika – FTIf ITS
Dosen Pembimbing I	: Prof. Drs. Ec. Ir. Riyanarto Sarno, M.Sc.,Ph.D.
Dosen Pembimbing II	: Bagus Setya Rintyarna S.T. M.Kom.

Abstrak

Pemeringkatan Perguruan Tinggi merupakan sebuah cara untuk mengukur kualitas dan kesiapan perguruan tinggi dalam menjalankan proses pembelajaran kepada mahasiswa yang dilihat dari berbagai macam kriteria penilaian.

Pemeringkatan Perguruan Tinggi menjadi salah satu hal yang penting untuk mengukur reputasi Perguruan Tinggi di dunia internasional maupun di dalam negeri. Dengan semakin baik reputasi, maka akan semakin menunjukkan bahwa kualitas pendidikan Perguruan Tinggi tersebut baik. Tidak hanya itu saja, melainkan juga menjadi cerminan kualitas pendidikan nasional dalam pandangan internasional.

Aspek kriteria penilaian bergantung pada berbagai macam penilaian yang dapat disimpulkan menjadi dua jenis yaitu kualitatif dan kuantitatif. Kriteria penilaian yang akan ditentukan harus mampu dijangkau oleh Perguruan Tinggi agar menghasilkan pemeringkatan yang seimbang.

Dalam Tugas Akhir ini diusulkan sebuah gagasan baru untuk sistem pemeringkatan Perguruan Tinggi di Indonesia menggunakan metode Helmholtz untuk mengoptimalkan kriteria penilaian yang bersifat kualitatif, yaitu mengoptimalkan kriteria

Reputasi Akademik dengan menggunakan topik modeling untuk menggali topik utama dari dokumen penelitian akademisi di setiap Perguruan Tinggi, yang sebelumnya dilakukan secara manual dengan melakukan survei oleh human. Pengoptimalan kriteria ini bertujuan agar menghasilkan hasil pemeringkatan yang lebih efisien dan sedernaha dari yang sebelumnya. Selain itu, untuk mengukur tingkat kesiapterapan teknologi dalam Perguruan Tinggi.

Hasil implementasi sistem pemeringkatan Perguruan Tinggi menunjukkan bahwa kebenaran sistem telah mendekati ground truth yaitu pemeringkatan QS University Rankings dengan tingkat akurasi 93%, dengan hasil ini menunjukkan bahwa sistem telah berhasil melakukan optimasi kriteria kualitatif Reputasi Akademik untuk mendekati hasil yang sudah ada. Dengan adanya sistem ini diharapkan proses pemeringkatan selanjutnya dapat mengacu pada konsep optimasi kriteria yang bersifat kualitatif agar menjadi lebih efisien dan sederhana dalam menghemat waktu kerja.

Kata kunci: Kualitas Perguruan Tinggi, Kualitatif, Kuantitatif, Helmholtz, Optimasi, Pemeringakatan Perguruan Tinggi, QS University Ranking, Reputasi

UNIVERSITY RANKING USING *HELMHOLTZ* METHOD TO MEASURE APPLICATION LEVEL TECHNOLOGY OF UNIVERSITY IN INDONESIA

Student Name : Anugerah Yulindra Satyaji
NRP : 5113100186
Major : Informatics Department – FTIf ITS
Supervisor I : Prof. Drs. Ec. Ir. Rivanarto Sarno, M.Sc.,Ph.D.
Supervisor II : Bagus Setya Rintyarna S.T. M.Kom.

Abstract

University ranking is one way to measure the quality and ability of college to apply their learning process with college student that seen from some criteria assessment.

University ranking has become one of the most important thing to measure college reputation in the outside world and within the country. A good reputation shows that college quality is good. Other than that, it can become a reflection of quality national education in the international views.

Aspect of assessment relies from a variety of assessment that can be summarized into two types, that is qualitative and quantitative. Assessment aspect specified needs to be able to be reached by the college in order to produce a balanced rankings.

This final project proposes a new idea for ranking university system in Indonesia using Helmholtz method to optimize the assessment criteria that are qualitative in order to generate the ranking to be more efficient and simpler. In addition, to measure the application level technology in university.

The result of the implementation of the university ranking system shows the correctness of the system has approached to ground truth that is the rank of QS University Ranking with the accuracy level of 93%. This result indicates the system has been successfully performed the optimization of qualitative criteria of

Academic Reputation to find the existing results. With this system, it is expected that further ranking process can be accepted with the concept of criteria optimization that is qualitative in order to be more efficient and simpler in saving working-time.

Keywords: Quality of University, Qualitative, Quantitative, Optimization, Ranking of University, QS University of Ranking, University, Reputation

KATA PENGANTAR

Puji syukur kehadiran Allah SWT, karena atas berkat rahmat-Nya, penulis dapat menyelesaikan Tugas Akhir yang berjudul:

“Pemeringkatan Perguruan Tinggi Menggunakan Metode Helmholtz Untuk Mengukur Tingkat Kesiapterapan Teknologi Perguruan Tinggi Di Indonesia”

Pengerjaan Tugas Akhir ini menjadi sebuah sarana untuk penulis memperdalam ilmu yang telah didapatkan selama menempuh pendidikan di kampus perjuangan Institut Teknologi Sepuluh Nopember Surabaya, khususnya dalam disiplin ilmu Teknik Informatika. Terselesaikannya buku Tugas Akhir ini tidak terlepas dari bantuan dan dukungan semua pihak. Pada kesempatan kali ini penulis ingin mengucapkan terima kasih kepada:

1. Bapak, Ibu, kakak, dan keluarga yang selalu memberikan dukungan penuh untuk menyelesaikan Tugas Akhir ini.
2. Bapak Riyanarto Sarno dan bapak Bagus Setya Rintyarna selaku dosen pembimbing yang telah bersedia meluangkan waktu untuk memberikan petunjuk selama proses pengerjaan Tugas Akhir ini.
3. Bapak dan Ibu dosen Jurusan Teknik Informatika ITS yang telah banyak memberikan ilmu dan bimbingan yang tak ternilai harganya bagi penulis.
4. Seluruh staf dan karyawan FTIf ITS yang banyak memberikan kelancaran administrasi akademik kepada penulis.
5. Segenap dosen rumpun mata kuliah Manajemen Informasi.
6. Rekan-rekan mahasiswa S2 Teknik Informatika yang telah banyak meluangkan waktu dan bersedia membantu penulis dalam memahami pengerjaan Tugas Akhir ini.

7. Rekan-rekan pengerjaan satu tim: Eko Putro Fitrianto, Donny Aliyanto, Arga Lancana dalam penyelesaian Tugas Akhir.
8. Teman-teman seperjuangan anak didik Tugas Akhir Prof. Rianarto Sarno yaitu Ahmad Zaenal Mustofa, Renanda Agustiantoro, Faisal Yanuar, Faisal Anugerah, Andi Putra, Hariyanto, Aldhiaz Fathra, Arfian Fidianoro, dan M. Fariz Ponighzwa.
9. Teman-teman Teknik Informatika Angkatan 2013 yang selalu mendukung, menyemangati, membantu dan mendengarkan suka duka selama proses pengerjaan Tugas Akhir.
10. Serta semua pihak yang turut membantu penulis dalam menyelesaikan Tugas Akhir ini.

Penulis menyadari bahwa Tugas Akhir ini masih memiliki banyak kekurangan. Dengan kerendahan hati, penulis mengharapkan kritik dan saran dari pembaca untuk perbaikan ke depan.

Surabaya, Juni 2017

Anugerah Yulindra Satyaji

DAFTAR ISI

LEMBAR PENGESAHAN.....	vii
Abstrak	ix
Abstract	xi
KATA PENGANTAR.....	xiii
DAFTAR ISI	xv
DAFTAR GAMBAR.....	xxi
DAFTAR TABEL	xxiii
DAFTAR KODE SUMBER.....	xxv
1 BAB I PENDAHULUAN	1
1.1 Latar Belakang	1
1.2 Rumusan Permasalahan.....	3
1.3 Batasan Permasalahan	3
1.4 Tujuan	4
1.5 Manfaat.....	5
1.6 Metodologi	5
1.7 Sistematika Penulisan.....	7
2 BAB II DASAR TEORI.....	9
2.1 University Rankings	9
2.2 Peringkat Universitas Dunia 4ICU.....	9
2.3 QS World University Rankings.....	10
2.4 Teks Mining	17
2.5 Teknik Grabbing	17

2.5.1	Fungsi cURL.....	18
2.5.2	Fungsi File_get_contents.....	19
2.5.3	Simple HTML Dom Parser.....	19
2.5.4	Regular Expression (ReGex)	20
2.6	Pusat Data Scopus	20
2.7	Google Scholar.....	21
2.8	Science and Technology Index (SINTA)	22
2.9	Pangkalan Data Pendidikan Tinggi (PDDIKTI)	23
2.10	Alexa Traffic Rank.....	24
2.11	MOZ Analytics.....	24
2.12	Bahasa Pemrograman Python.....	25
2.13	Library Python	26
2.14	PHP dan HTML	30
2.15	Database Management System (DBMS) MySQL.....	31
2.16	Part-Of-Speech Tagger (POS Tagger)	31
2.17	Bloom Taxonomy.....	33
2.18	Helmholtz.....	34
2.19	Labeled Latent Dirichlet Allocation (L-LDA)	36
2.20	Term Frequency – Invers Document Frequency (TF-IDF).42	
2.20.1	Term Frequency (TF)	42
2.20.2	Invers Document Frequency (IDF)	43
2.21	Akreditasi Perguruan Tinggi	43
2.22	Tingkat Kesiapterapan Teknologi (TKT).....	44
3	BAB III ANALISIS DAN PERANCANGAN SISTEM	47
3.1.	Tahap Analisis.....	47
3.1.1	Deskripsi Umum	47

3.1.2	Arsitektur Perangkat Lunak	47
3.1.3	Analisis Aktor	48
3.1.4	Analisis Permasalahan	48
3.1.5	Spesifikasi Kebutuhan Perangkat Lunak	48
3.1.6	Kasus Penggunaan	49
3.1.7	Label Kelas dalam Teks Mining Metode L-LDA	60
3.1.8	Ground Truth Pengujian Sistem	61
3.1.9	Dataset Pengujian Sistem	65
3.2.	Tahap Perancangan Sistem	69
3.2.1	Perancangan Basis Data	69
3.2.2	Perancangan Antarmuka	82
3.2.3	Perancangan Antarmuka	84
3.2.4	Proses Sistem Aplikasi Pengguna (Diagram Aplikasi)	90
3.2.5	Proses Kerja Sistem Secara Keseluruhan (Flowchart Sistem)	91
3.2.6	Proses Teks Mining Metode Helmholtz (Flowchart Metode Helmholtz)	92
4	BAB IV IMPLEMENTASI SISTEM	93
4.1	Lingkungan Implementasi	93
4.2	Tahap Implementasi	93
4.2.1	Tahap Prapemrosesan	93
4.2.2	Memperkaya Corpus	94
4.3	Tahap Fitur Seleksi TKT	94
4.4	Klasifikasi Term Frequency	95
4.5	Penentuan Nilai Akhir Peringkat Universitas	95

4.6	Implementasi Basis Data.....	97
4.6.1	Create Database “databse_qs_rank”	97
4.6.2	Create Table “universitas”	97
4.6.3	Create Table “reputasi_akademik”	97
4.6.4	Create Tabel “akreditasi”	98
4.6.5	Create Tabel “jumlah_tenaga_pengajar_jabatan_akademik”	98
4.6.6	Create Tabel “jumlah_mahasiswa”	99
4.6.7	Create Tabel “jumlah_tenaga_pengajar_tetap”	99
4.6.8	Create Tabel “jumlah_tenaga_pengajar_tidak_tetap”	100
4.6.9	Create Tabel “jumlah_sitasi_sinta”	101
4.6.10	Create Tabel “jumlah_author_sinta”	101
4.6.11	Create Tabel “presence”	102
4.6.12	Create Tabel “impact”	102
4.6.13	Create Tabel “alexa_rank”	102
4.6.14	Create Tabel “kumpulan_skor_final”	103
4.6.15	Create Tabel “4icu”	104
4.6.16	Create Tabel “dikti”	104
4.6.17	Create Tabel “qs_general”	104
4.6.18	Create Tabel “qs_asia”	105
4.6.19	Create Tabel “qs_helmholtz”	105
4.6.20	Create Tabel “webometrics”	105
4.7	Antarmuka Halaman Utama.....	106
4.7.1	Antarmuka Kolom Search	106
4.7.2	Antarmuka Peringkat Berdasarkan Region	106

4.7.3	Antarmuka Atur Bobot Pemeringkatan	107
4.7.4	Antarmuka Daftar Peringkat Universitas	107
4.7.5	Antarmuka Rincian Universitas	108
5	BAB V PENGUJIAN DAN EVALUASI	109
5.1	Lingkungan Uji Coba.....	109
5.2	Data Uji Coba.....	109
5.2.1	Data Dokumen Abstrak Paper Akademisi	110
5.2.2	Data Dokumen Abstrak Paper Akademisi	110
5.2.3	Data Jumlah Mahasiswa dan Tenaga Pengajar	111
5.2.4	Data Online Science and Technologu Index (SINTA)	111
5.2.5	Data Statistik dan Analitik Online	113
5.3	Skenario Pengujian.....	114
5.3.1	Prapemrosesan.....	114
5.3.2	Memperkaya Corpus TKT	115
5.3.3	Fitur Seleksi Corpus TKT	116
5.3.4	Pembobotan Metode Generatif Model L-LDA	116
5.4	Evaluasi Pengujian	125
6	BAB VI KESIMPULAN DAN SARAN.....	126
6.1	Kesimpulan.....	127
6.2	Saran.....	127
	DAFTAR PUSTAKA.....	129
	LAMPIRAN	133
	BIODATA PENULIS.....	185

[Halaman ini sengaja dikosongkan]

DAFTAR GAMBAR

Gambar 2.1 Proses Utama Teks Mining.....	17
Gambar 2.2 Diagram Representasi Model Generatif LDA	37
Gambar 2.3 Diagram Model Generatif LDA.....	41
Gambar 2.4 Hirarki Tingkat Kesiapterapan Teknologi (TKT)...	45
Gambar 3.1 Diagram Kasus Penggunaan	50
Gambar 3.2 Diagram Aktivitas UC-001	52
Gambar 3.3 Diagram Aktivitas UC-002.....	54
Gambar 3.4 Diagram Aktivitas UC-003.....	56
Gambar 3.5 Diagram Aktivitas UC-004.....	57
Gambar 3.6 Diagram Aktivitas UC-005.....	59
Gambar 3.7 Conceptual Data Model	80
Gambar 3.8 Physical Data Model.....	81
Gambar 3.9 Alur Menentukan Reputasi Akademik	85
Gambar 3.10 Alur Menentukan Skor Akreditasi.....	86
Gambar 3.11 Diagram Aplikasi.....	90
Gambar 3.12 Flowchart Sistem Keseluruhan	91
Gambar 3.13 Flowchart Metode Helmholtz	92
Gambar 4.1 Output Helmholtz	95
Gambar 4.2 Antarmuka Kolom Search	106
Gambar 4.3 Antarmuka Peringkat Berdasarkan Region.....	107
Gambar 4.4 Antarmuka Atur Bobot Pemingkatan	107
Gambar 4.5 Antarmuka Daftar Peringkat Universitas.....	108
Gambar 4.6 Antarmuka Rincian Universitas.....	108

[Halaman ini sengaja dikosongkan]

DAFTAR TABEL

Tabel 2.1 Struktur Kelas kata POS-Tagging	32
Tabel 3.1 Daftar Kebutuhan Fungsional Sistem.....	48
Tabel 3.2 Keterangan Kode Kasus Penggunaan.....	50
Tabel 3.3 Spesifikasi Kasus Melihat Peringkat Universitas.....	51
Tabel 3.4 Spesifikasi Kasus Memasukkan Bobot Kriteria	52
Tabel 3.5 Spesifikasi Kasus Penggunaan Mencari Nama Universitas.....	55
Tabel 3.6 Spesifikasi Kasus Penggunaan Mencari Wilayah Provinsi Universitas.....	56
Tabel 3.7 Spesifikasi Kasus Penggunaan Melihat Rincian Informasi Universitas	58
Tabel 3.8 Penjelasan Setiap Kelas Label.....	60
Tabel 3.9 Peringkat Reputasi Akademik QS 2015-2016.....	61
Tabel 3.10 Peringkat QS World University Rankings 2015-2016	62
Tabel 3.11 Peringkat Reputasi Akademik QS 2016-2017.....	63
Tabel 3.12 Peringkat QS World University Rankings 2016-2017	63
Tabel 3.13 Peringkat Reputasi Akademik QS 2017-2018.....	64
Tabel 3.14 Peringkat QS World University Rankings 2017-2018	64
Tabel 3.15 Contoh Hasil Analisa Manual Sebagai Ground Truth Reputasi Akademik	65
Tabel 3.16 Daftar Rincian Jumlah Kata Dataset Abstrak Paper Perguruan Tinggi	67
Tabel 3.17 Daftar Rincian Jumlah Kata Corpus Taxonomy Bloom	68
Tabel 3.18 Perbandingan Jumlah Kata Corpus TKT.....	69
Tabel 4.1 Bobot Level TKT	96
Tabel 5.1 Gabungan Jumlah Kata Taxonomy Bloom	110
Tabel 5.2 Jumlah Kata Hasil Pemrosesan	114
Tabel 5.3 Hasil Akuisisi POS-Tagging 1	115
Tabel 5.4 Hasil Fitur Seleksi Helmholtz	116
Tabel 5.5 Nilai Akhir Universitas POS-Tagging.....	117

Tabel 5.6 Peringkat Universitas POS-Tagging	118
Tabel 5.7 Nilai Akhir Universitas WordNet level 1	118
Tabel 5.8 Peringkat Universitas WordNet level 1	119
Tabel 5.9 Nilai Akhir Universitas WordNet level 2	119
Tabel 5.10 Peringkat Universitas WordNet level 2	119
Tabel 5.11 Nilai Akhir Universitas WordNet level 3	120
Tabel 5.12 Peringkat Universitas WordNet level 3	120
Tabel 5.13 Nilai Akhir Universitas WordNet level all	121
Tabel 5.14 Peringkat Universitas WordNet All	121
Tabel 5.15 Perbandingan Hasil POS-Tagging	122
Tabel 5.16 Perbandingan Hasil WordNet Level 1	122
Tabel 5.17 Perbandingan Hasil Wordnet Level 2	122
Tabel 5.18 Perbandingan Hasil Wordnet Level 3	123
Tabel 5.19 Perbandingan Hasil Wordnet Level All	123

DAFTAR KODE SUMBER

Kode Sumber 1 Syntax cURL dengan setting option url	18
Kode Sumber 2 Syntax Fungsi file_get_contents.....	19
Kode Sumber 3 Syntax Penggunaan Library Simple HTML Dom Parser.....	19
Kode Sumber 2.1 Metode Helmholtz.....	36
Kode Sumber 5 Kode Sumber Create Database.....	97
Kode Sumber 6 Kode Sumber Create Table universitas	97
Kode Sumber 7 Kode Sumber Create Table reputasi_akademik	97
Kode Sumber 8 Kode Sumber Create Table akreditasi.....	98
Kode Sumber 9 Kode Sumber Create Table jumlah_tenaga_pengajar_jabatan_akademik.....	98
Kode Sumber 10 Kode Sumber Create Table jumlah_mahasiswa	99
Kode Sumber 11 Kode Sumber Create Table jumlah_tenaga_pengajar_tetap.....	99
Kode Sumber 12 Kode Sumber Create Table jumlah_tenaga_pengajar_tidak_tetap.....	100
Kode Sumber 13 Kode Sumber Create Table jumlah_sitasi_sinta	101
Kode Sumber 14 Kode Sumber Create Table jumlah_author_sinta	101
Kode Sumber 15 Kode Sumber Create Table presence.....	102
Kode Sumber 16 Kode Sumber Create Table impact.....	102
Kode Sumber 17 Kode Sumber Create Table alexa_rank	102
Kode Sumber 18 Kode Sumber Create Table kumpulan_skor_final.....	103
Kode Sumber 19 Kode Sumber Create Table 4icu.....	104
Kode Sumber 20 Kode Sumber Create Table dikti	104
Kode Sumber 21 Kode Sumber Create Table qs_general	104
Kode Sumber 22 Kode Sumber Create Table qs_asia.....	105
Kode Sumber 23 Kode Sumber Create Table qs_helmholtz.....	105
Kode Sumber 24 Kode Sumber Create Table webometrics	105
Kode Sumber 2 Preprocessing & Labeling Labeling	158

Kode Sumber 3 Helmholtz	163
Kode Sumber 4 Term Frequency	171
Kode Sumber 5 Labeled Latent Dirichlet Allocation	181
Kode Sumber 6 Pembobotan Dokumen	184

BAB I

PENDAHULUAN

Pada bab ini akan dipaparkan mengenai garis besar Tugas Akhir yang meliputi latar belakang, tujuan, rumusan dan batasan permasalahan, metodologi pengerjaan Tugas Akhir, dan sistematika penulisan Tugas Akhir.

1.1 Latar Belakang

Perguruan Tinggi merupakan sebuah institusi pendidikan penyelenggara pendidikan tinggi. Pendidikan tinggi menjadi hal yang sangat penting dikarenakan mencerminkan kondisi intelektual masyarakat dari sebuah negara. Membutuhkan perhatian dan evaluasi lebih dari pemerintah untuk dapat menghasilkan sumberdaya manusia yang baik dari Perguruan Tinggi. Hal ini tentunya untuk menghasilkan generasi yang mempunyai intelektual dan keterampilan tinggi ketika bersaing di dunia kerja nantinya. Kualitas pendidikan tinggi yang baik menjadi prioritas dan pusat perhatian pendidikan internasional.

Pencapaian tujuan strategis Perguruan Tinggi bergantung pada faktor kontekstual, yang meliputi kepemimpinan suatu negara, ideologi negara, finansial, sosial budaya, komunikasi dan bahasa, serta proses pengambilan keputusan. Selain itu Perguruan Tinggi juga didorong oleh norma yang terkait dengan lingkungan dan nilai pendidikan itu sendiri.

Di sisi lain, Perguruan Tinggi merupakan salah satu sarana yang efektif untuk mengembangkan ilmu pengetahuan dan teknologi dengan berbagai cara, salah satu cara dengan melakukan penelitian. Penelitian merupakan salah satu wadah kreativitas dan pengembangan potensi akademik bagi mahasiswa di sebuah negara. Selain sebagai wadah kreativitas dan pengembangan potnsi akademik, penelitian diselenggarakan oleh Perguruan Tinggi bertujuan untuk membuat seorang akademisi dapat menghasilkan karya atau ciptaan dalam dunia pendidikan. Selain itu jumlah populasi masyarakat suatu negara yang banyak maka harus berbanding lurus dengan pengembangan intelektual

dan keterampilan generasi mudanya, sebagai investasi pemerintah dalam dunia pendidikan.

Salah satu upaya untuk meningkatkan kualitas Perguruan Tinggi adalah dengan melakukan pemeringkatan Perguruan tinggi di seluruh dunia. Setiap Perguruan Tinggi yang masuk dalam peringkat dunia akan menjadi acuan bahwa kualitas pendidikan di negara tersebut bagus, baik dalam kriteria apapun yang dibungkus dengan satu wadah utama yaitu nama Perguruan Tinggi. Ada banyak media publikasi pemeringkatan Perguruan Tinggi, salah satunya adalah publikasi tahunan Perguruan Tinggi QS University Rankings. Dalam ranking Perguruan Tinggi, tentunya ada banyak kriteria penilaian yang menjadikan Perguruan Tinggi tersebut berkualitas dan layak untuk menempati peringkat dalam daftar.

Tingkat kualitas pendidikan di Indonesia sejauh ini masih dapat dikatakan kurang baik. Indonesia masih berfokus pada jumlah partisipasi pendidikan, tidak pada pembangunan kualitas pendidikan. Di sisi lain, Perguruan Tinggi di Indonesia juga mempunyai perhatian baik dari dunia internasional, dikarenakan ada beberapa karya penelitian dari Perguruan Tinggi Indonesia yang bermanfaat dan dikenal oleh dunia internasional. Berdasarkan posisi peringkat publikasi tahunan QS University Rankings, Perguruan Tinggi Indonesia masih berada pada urutan 67 regional Asia yang ditempati oleh Universitas Indonesia (UI). Berbeda jauh jika dibandingkan dengan negara tetangga, yaitu Singapura yang notabene jumlah penduduknya lebih sedikit dari Indonesia berhasil berada pada peringkat 1 region Asia yang ditempati oleh National University of Singapore (NUS) (2). Di Indonesia sendiri kualitas pendidikan yang menonjol hanya berada pada Perguruan Tinggi tertentu saja, hal ini yang menyebabkan kualitas pendidikan di Indonesia kurang merata.

Salah satu cara untuk mengetahui merata atau tidaknya kualitas pendidikan tinggi di Indonesia dengan melihat Tingkat kesiapterapan teknologi (TKT) dari Perguruan Tinggi. TKT ini menggambarkan mengenai tingkat kematangan Perguruan Tinggi dalam hal penelitian dan pengembangan teknologi. Tujuan dari

penelitian secara umum sejalan dengan tujuan Perguruan Tinggi sebagai sarana yang efektif untuk mengembangkan ilmu pengetahuan dan pengembangan potensi sumberdaya manusia (mahasiswa).

1.2 Rumusan Permasalahan

Rumusan masalah yang diangkat dalam Tugas Akhir ini dapat dipaparkan sebagai berikut:

1. Bagaimana mengumpulkan data masukan sebagai bahan penilaian kriteria perguruan tinggi?
2. Bagaimana menghasilkan skor akhir dari kriteria penilaian perguruan tinggi yang bersifat kualitatif menggunakan fitur ekstraksi metode *Helmholtz* dengan bantuan *Labeled Latent Dirichlet Allocation* (L-LDA) sebagai proses topik modelling?
3. Bagaimana menghitung skor akhir dari data kriteria penilaian perguruan tinggi yang bersifat kuantitatif?
4. Bagaimana menentukan skor akhir secara keseluruhan untuk sebuah perguruan tinggi?
5. Bagaimana menganalisa hasil keluaran sistem dengan *Ground Truth QS University Rankings* yang sudah ditetapkan sehingga menghasilkan hasil akurasi?

1.3 Batasan Permasalahan

Permasalahan yang dibahas dalam Tugas Akhir ini memiliki beberapa batasan, di antaranya sebagai berikut:

1. Bahasa pemrograman yang digunakan adalah Python untuk proses *teks mining* dokumen abstrak *paper*, PHP dan HTML untuk membangun sistem *grabbing* dan membangun desain antarmuka.
2. Database management sistem (DBMS) yang digunakan adalah MySQL database.
3. Perguruan tinggi yang dinilai hanya perguruan tinggi dalam wilayah negara Indonesia.

4. Menggunakan acuan kriteria dari media publikasi tahunan *QS World University Rankings*.
5. Data sebagai dasar daftar perguruan tinggi, data total mahasiswa, data total tenaga kerja perguruan tinggi berdasarkan data pada PDDIKTI.
6. Data total publikasi akademisi sebagai salah satu kriteria penilaian kuantitatif berdasarkan database *Science and Technology Index* (SINTA) yang mengadopsi database dari Scopus dan Google Scholar.
7. Data akreditasi perguruan tinggi berdasarkan data dari Badan Akreditasi Nasional Perguruan Tinggi (BAN-PT).
8. Metode *Helmholtz* dibantu dengan *Labeled Latent Dirichlet Allocation* (L-LDA) untuk mengidentifikasi topik dari dokumen abstrak *paper* akademisi perguruan tinggi yang akan menjadi skor nilai reputasi akademik yang bersifat kualitatif.
9. Menggunakan 50 dataset berupa dokumen abstrak *paper* akademisi dari masing-masing perguruan tinggi dengan kategori rumpun teknik.
10. Menggunakan corpus *Taxonomy Bloom* sebagai dasar penetapan corpus kata (*term*) untuk setiap level Tingkat Kesiapterapan Teknologi (TKT).

1.4 Tujuan

Tujuan dari Tugas Akhir ini adalah sebagai berikut:

1. Mampu menerapkan teknik *grabbing* terhadap data online berupa informasi *numeric* sebagai masukan kriteria penilaian perguruan tinggi yang bersifat kuantitatif.
2. Mampu menerapkan metode *Helmholtz* dibantu dengan *Labeled Latent Dirichlet Allocation* (L-LDA) untuk memodelkan topik dari sebuah dokumen abstrak *paper* akademisi perguruan tinggi sebagai masukan kriteria penilaian yang bersifat kualitatif.

3. Mampu menghasilkan skor dari setiap kriteria penilaian perguruan tinggi, serta skor final penilaian secara keseluruhan/kumulatif dari setiap kriteria.
4. Mampu menganalisa hasil keluaran sistem pemeringkatan dengan *Ground Truth QS World University Rankings*.
5. Mampu menyajikan informasi hasil pemeringkatan perguruan tinggi dalam antarmuka yang jelas dan mudah untuk dimengerti dan diakses oleh pengguna perangkat lunak website (*End User*).

1.5 Manfaat

Tugas Akhir ini diharapkan dapat membantu menghasilkan pemeringkatan perguruan tinggi di Indonesia yang mudah, efektif, dan sederhana berdasarkan Tingkat Kesiapterapan Teknologi (TKT) yang menjadi dasar pemetaan tingkat kematangan teknologi perguruan tinggi di Indonesia. Mengoptimasi kriteria diharapkan mampu menekan waktu kerja (*runnning time*) dan biaya (*cost*) untuk proses pemeringkatan perguruan tinggi.

Penilaian kriteria yang dibahas mengacu pada publikasi pemeringkatan perguruan tinggi internasional tahunan *QS World University Rankings* yang dapat digunakan sebagai tolok ukur kualitas pendidikan tinggi di Indonesia.

1.6 Metodologi

Langkah-langkah yang ditempuh dalam pengerjaan Tugas Akhir ini yaitu:

1. Studi literatur

Pada tahap ini dilakukan studi literatur dengan mencari referensi yang berfokus pada pembahasan teknik *grabbing* data online dari website yang berfungsi sebagai penyedia informasi kriteria penilaian yang bersifat kuantitatif, mencari referensi proses teks mining metode *Helmholtz* untuk proses fitur ekstraksi dan *Labeled Latent Dirichlet Allocation* (L-LDA) tdokumen abstrak *paper* akademisi perguruan tinggi yang menjadi salah satu

kriteria penilaian yang mempunyai bobot ketergantungan paling tinggi, serta mencari referensi literatur lain yang membahas mengenai dasar-dasar dan proses pemeringkatan perguruan tinggi yang sudah ada sebelumnya.

2. Analisis dan Perancangan Sistem

Aktor yang menjadi pelaku adalah pengguna perangkat lunak yang dibangun oleh penulis. Kemudian beberapa kebutuhan fungsional dari sistem ini adalah sebagai berikut:

- a. Analisis aktor yang terlibat di dalam sistem.
- b. Perancangan diagram *use case*, yang merupakan analisis kebutuhan pada aplikasi yang akan dibangun.
- c. Analisis kebutuhan non-fungsional.
- d. Perancangan sistem pemeringkatan universitas.
- e. Perancangan metode *Helmholtz* beserta *Labeled Latent Dirichlet Allocation* (L-LDA).

3. Implementasi

Pada tahap ini dilakukan pembuatan perangkat lunak untuk sistem server (*developer*) berupa penerapan teknik *grabbing* pada pengambilan data online, proses teks mining menggunakan metode *Helmholtz* dibantu dengan *Labeled Latent Dirichlet Allocation* (L-LDA) untuk pengganti kriteria reputasi akademik yang sebelumnya dikerjakan secara manual, serta membangun database dan antarmuka untuk menampilkan informasi akhir kepada pengguna.

4. Pengujian dan evaluasi

Pada tahap ini dilakukan pengujian terhadap perangkat lunak dengan menggunakan dataset final berupa 50 dokumen abstrak *paper* akademisi dari masing-masing perguruan tinggi sebagai pengujian kriteria penilaian yang sifat kualitatif, kemudian menggunakan data online dari database akademisi *Science and*

Technology Index (SINTA) sebagai pengujian kriteria penilaian yang bersifat kuantitatif yang mempunyai nilai berupa numeric, serta menggunakan data hasil analisa Backlink, Domain Authority dan Page Authority dari kriteria berbasis analisa *webhost*.

Uji coba dan evaluasi dilakukan untuk mengevaluasi jalannya perangkat lunak, mengevaluasi fitur utama, mengevaluasi fitur tambahan, mencari kesalahan antara hasil peringkat yang dihasilkan oleh sistem perangkat lunak dengan peringkat *Ground Truth* yang sudah ditetapkan, menganalisa faktor-faktor yang mempengaruhi kebenaran sistem, dan mengadakan perbaikan jika ada kekurangan.

5. Penyusunan buku Tugas Akhir

Pada tahap ini dilakukan pendokumentasian dan pelaporan dari seluruh konsep, dasar teori, implementasi, proses yang telah dilakukan, dan hasil-hasil yang telah didapatkan selama pengerjaan Tugas Akhir.

1.7 Sistematika Penulisan

Buku Tugas Akhir ini bertujuan untuk mendapatkan gambaran dari pengerjaan Tugas Akhir ini. Selain itu, diharapkan dapat berguna untuk pembaca yang tertarik untuk melakukan pengembangan lebih lanjut. Secara garis besar, buku Tugas Akhir terdiri atas beberapa bagian seperti berikut ini.

Bab I Pendahuluan

Bab ini berisi latar belakang masalah, tujuan dan manfaat pembuatan Tugas Akhir, permasalahan, batasan masalah, metodologi yang digunakan, dan sistematika penyusunan Tugas Akhir.

Bab II Dasar Teori

Bab ini membahas beberapa teori penunjang yang berhubungan dengan pokok pembahasan dan yang menjadi dasar dari pembuatan Tugas Akhir ini.

Bab III Analisis dan Perancangan Sistem

Bab ini membahas mengenai perancangan perangkat lunak. Perancandaftagan perangkat lunak meliputi perancangan alur, proses dan perancangan antarmuka pada perangkat lunak.

Bab IV Implementasi

Bab ini berisi implementasi dari perancangan perangkat lunak perangkat lunak dan implementasi fitur-fitur penunjang perangkat lunak.

Bab V Pengujian dan Evaluasi

Bab ini membahas pengujian dengan metode pengujian subjektif untuk mengetahui penilaian aspek kegunaan (*usability*) dari perangkat lunak dan pengujian fungsionalitas yang dibuat dengan memperhatikan keluaran yang dihasilkan serta evaluasi terhadap fitur-fitur perangkat lunak.

Bab VI Kesimpulan

Bab ini berisi kesimpulan dari hasil pengujian yang dilakukan. Bab ini membahas saran-saran untuk pengembangan sistem lebih lanjut.

Bab VII Daftar Pustaka

Merupakan daftar referensi yang digunakan untuk mengembangkan Tugas Akhir.

BAB II

DASAR TEORI

Pada bab ini akan dibahas mengenai teori-teori yang menjadi dasar dari pembuatan Tugas Akhir.

2.1 *University Rankings*

Peringkat perguruan tinggi merupakan salah satu cara untuk mengukur kualitas pendidikan di sebuah negara yang diterapkan oleh seluruh perguruan tinggi di dunia. Namun saat ini tujuan pemeringkatan perguruan tinggi cenderung berbeda dari awalnya yaitu menjadi salah satu cara untuk melihat kepopuleran perguruan tinggi tersebut di mata dunia internasional. Dimana perguruan tinggi yang mempunyai posisi peringkat tinggi akan menjadi perhatian bahwa pendidikan tinggi di negara tersebut berkualitas [1]- [2].

Hingga saat ini, ada berbagai macam publikasi pemeringkatan perguruan tinggi yang populer untuk memberikan hasil pemeringkatan perguruan tinggi di seluruh dunia [3]- [8], diantaranya:

1. QS World University Rankings.
2. Timer Higher Education World University Rankings (THE).
3. Academics Ranking of World University (ARWU).
4. Global Universities Rangkaing by US News.
5. Reuters World's Top 100 Innovative University.
6. SCImago Institution Rankings.
7. Webometrics.
8. 4ICU (4 International Colleges and Universities).

2.2 Peringkat Universitas Dunia 4ICU

Merupakan indeks pencarian serta peringkat universitas/organisasi yang memiliki kriteria sebagai berikut: terakreditasi secara resmi oleh kementerian pendidikan di negara tempat universitas/organisasi didirikan, memiliki program empat tahun strata satu atau program strata dua, menyediakan kelas,

program dan kuliah tatap-muka tradisional. 4ICU memiliki tata cara/algorithm penilaian dengan mengandalkan Google Page Rank, Alexa Page Rank, serta Majestic SEO. [2]

2.3 QS World University Rankings

Quacquarelli Symonds atau lebih sering disebut dengan *QS World University Ranking* merupakan publikasi peringkat perguruan tinggi seluruh dunia yang berlokasi di Inggris [4]. *QS World University Ranking* diterbitkan setiap tahun dengan menghasilkan perubahan peringkat yang sangat sensitif, sehingga persaingan antar perguruan tinggi begitu kuat [1].

Awal mula terbentuknya *QS World University Rankings* pada tahun 2004 merupakan hasil kolaborasi dengan THE (*Times Higher Education*) dengan nama publik THE-QS, namun sejak tahun 2009 QS dan THE sepakat untuk berjalan sendiri-sendiri. QS berjalan hingga sekarang menggunakan metode yang diterapkan sejak awal dan lebih populer disebut publikasi *QS World University Ranking*. Sedangkan THE membangun publikasi peringkat perguruan tinggi sendiri dengan nama THES yang menggunakan metode lain yang hampir serupa dengan *QS World University Ranking* [3].

QS World University Ranking mempunyai 5 cakupan region sebagai observasi pemeringkatan, diantaranya Asia, Amerika Latin, negara Eropa berkembang - Asia Tengah, region Arab dan BRICS (Brasil, Rusia, India, China, South Africa). Untuk menghasilkan peringkat perguruan tinggi, *QS World University Ranking* mempunyai 6 kriteria utama [4], diantaranya:

1. Reputasi Akademik (40%)

Kriteria ini merupakan hasil survey responden akademisi perguruan tinggi di seluruh dunia. Survey ini diharapkan akan mendapatkan hasil yang objektif dari semua responden yang tersebar diseluruh dunia. Mereka mendapatkan bagian survey sesuai dengan latar belakang akademisi tersebut. Terdapat 5 garis besar bidang latar belakang akademisi sebagai responden, yaitu bidang ilmu

seni dan sastra, ilmu teknik dan teknologi, ilmu biologi dan kesehatan, ilmu alam, dan ilmu sosial [3].

2. Reputasi Employer (10%)

Kriteria ini merupakan hasil survey responden perusahaan yang menaungi alumni perguruan tinggi di seluruh dunia [3]. Sama dengan kriteria reputasi akademik, kriteria ini ditujukan agar memperoleh hasil yang objektif terhadap kualitas lulusan (fresh graduate) perguruan tinggi. Salah satu faktor penting agar perguruan tinggi dapat diperhatikan dalam kriteria ini adalah dengan menjalin banyak kerjasama dengan perusahaan sebagai rekan kerja maupun penelitian [2].

3. Jumlah Mahasiswa Aktif (20%)

Kriteria ini merupakan hasil jumlah mahasiswa yang sedang menempuh pendidikan di perguruan tinggi. Tujuan dari kriteria ini adalah untuk menganalisa pengawasan jumlah infrastruktur dengan jumlah mahasiswa, agar jumlah infrastruktur sebanding dengan jumlah mahasiswa. Perguruan tinggi yang baik merupakan perguruan tinggi yang memberikan fasilitas dan kenyamanan belajar [3].

4. Jumlah Sitasi Akademisi (20%)

Kriteria ini merupakan hasil jumlah sitasi maupun publikasi dari akademisi perguruan tinggi yang ter-*index* oleh publikasi Scopus. Sitasi/publikasi yang dihitung berusia maksimal terbit 5 tahun sebelumnya. Tujuan dari kriteria ini adalah untuk melihat seberapa banyak dampak penelitian akademisi perguruan tinggi terhadap lingkungan [3]. Karena tujuan perguruan tinggi adalah yang dapat memberikan dampak baru bagi masyarakat.

5. Jumlah Tenaga Pengajar Internasional (5%)

Kriteria ini merupakan hasil dari jumlah tenaga pengajar (staff/dosen) asing di sebuah perguruan tinggi. Lama waktu

minimal tenaga pengajar asing menetap selama 3 bulan per tahun [2]. Tujuan dari kriteria ini adalah melihat kerjasama dan ketertarikan tenaga asing untuk perguruan tinggi tersebut.

6. Jumlah Mahasiswa Asing (5%)

Kriteria ini merupakan hasil dari jumlah mahasiswa asing yang belajar di perguruan tinggi, baik dalam rangka pertukaran pelajar, maupun program pendidikan lainnya [3]. Tujuan dari kriteria ini adalah untuk melihat ketertarikan mahasiswa asing terhadap metode pembelajaran perguruan tinggi [2].

Dalam penilaian kriteria *QS World University Ranking* terdapat 2 jenis kriteria, yaitu kriteria yang bersifat kualitatif dan kuantitatif. Untuk kriteria yang bersifat kualitatif merupakan kriteria yang membutuhkan pengolahan data untuk mendapatkan nilai serta cenderung bersifat deskriptif dan membutuhkan analisa terlebih dahulu. Kriteria yang termasuk dalam jenis kualitatif adalah kriteria reputasi akademik dan reputasi employer.

Sedangkan untuk kriteria yang bersifat kuantitatif merupakan kriteria yang dapat ditentukan nilainya secara langsung tanpa melakukan analisa terlebih dahulu dan tidak bersifat deskriptif. Kriteria yang termasuk dalam jenis kuantitatif adalah kriteria jumlah mahasiswa aktif, jumlah sitasi/publikasi akademisi, jumlah staff internasional dan jumlah mahasiswa asing.

Untuk kriteria pertama yang bersifat kualitatif, yaitu reputasi akademik dalam *QS World University Rankings* ada beberapa poin yang menjadi acuan untuk digali dari responden survey untuk menghasilkan penilaian yang objektif [9], di antaranya:

1. Biodata personal responden akademisi

Poin penilaian ini, responden diminta menjelaskan mengenai biodata personal responden yang berkaitan dengan dimana akademisi bekerja, diantaranya seperti

nama lengkap, alamat email institusional, jobdesk pekerjaan, lama bekerja dalam institusi, nama institusi dimana akademisi bekerja, pendapat mengenai negara yang paling berpengaruh dalam pendidikan tinggi dan penelitian, serta menanyakan mengenai pengetahuan akademisi terhadap Massive Open Online Courses (MOOCs) [9].

2. Knowledge specification

Poin penilaian ini, responden diminta menjelaskan berdasar pengetahuan responden mengenai pengetahuan dasar tentang region dan bidang ilmu yang akan dibahas/ditanyakan, diantaranya seperti pengetahuan terhadap kualitas akademik berdasarkan region (amerika, asia-australia-new zealand, eropa-timur tengah-afrika atau semua region), pengetahuan terhadap kategori bidang ilmu pengajaran, pengetahuan terhadap sub bidang ilmu yang lebih spesifik, serta mengenai pembagian waktu akademisi untuk melakukan riset (*research*), pengajaran (*teaching*), administrasi (*administration*) dan lainnya dalam bentuk persen [9].

3. Penelitian terbaik

Poin penilaian ini, responden diminta memilih beberapa perguruan tinggi yang dianggap mempunyai tingkat penelitian (*research*) yang baik, diantaranya seperti memilih 10 institusi dalam negeri sesuai dengan region negara responden yang mempunyai kualitas penelitian terbaik, memilih 30 institusi luar negeri yang mempunyai kualitas penelitian terbaik, serta dalam memilih institusi dalam negeri maupun luar negeri tersebut harus mencakup kategori bidang ilmu yang diteliti. Ada 5 bidang ilmu, maka total responden harus memilih 50 perguruan tinggi dalam negeri terbaik yang mencakup 5 bidang ilmu, serta 150 perguruan luar negeri yang mencakup 5 bidang ilmu.

4. Inovasi dalam penelitian

Poin penilaian ini, responden diminta mengidentifikasi perguruan tinggi mana yang mempunyai tingkat inovasi yang baik dalam penelitian, diantaranya seperti mengidentifikasi institusi yang paling cepat dan dinamis dalam perkembangan dan inovasi penelitian, serta bidang ilmu apa yang dimiliki institusi tersebut yang paling cepat berkembang dalam hal inovasi [9].

5. Leading company

Poin penilaian ini, responden diminta mengidentifikasi perusahaan mana yang paling berkontribusi terhadap penelitian, diantaranya seperti mengidentifikasi 5 perusahaan yang menurut responden akademisi paling berpengaruh terhadap pendidikan sesuai dengan bidang pendidikan responden, serta menjelaskan alasan memilih 5 perusahaan tersebut [9].

6. MOOCs

Poin penilaian ini, responden diminta menjelaskan mengenai penerapan MOOCs dalam institusi responden [10], diantaranya seperti apakah institusi pendidikan responden sudah menerapkan pembelajaran berbasis MOOCs, apakah institusi pendidikan responden dapat merencanakan pembelajaran berbasis MOOCs untuk 3 tahun kedepan, apakah responden telah menerapkan pembelajaran MOOCs berdasarkan inisiatif pribadi, serta pendapat responden mengenai keuntungan pembelajaran berbasis MOOCs [9].

7. Profil penelitian responden akademisi

Poin penilaian ini, responden diminta menjelaskan profil publikasi/penelitian responden selama ini jika ada, diantaranya seperti artikel jurnal, konferensi, paper/buku, kemampuan (*performance*) responden, pengalaman (*exhibitions*) responden, serta paten yang dimiliki [9].

Untuk kriteria kedua yang bersifat kualitatif yaitu reputasi employer, ada beberapa poin yang menjadi acuan untuk digali dari responden perusahaan untuk menghasilkan penilaian yang objektif [11], di antaranya:

1. Biodata personal responden perusahaan

Poin penilaian ini, responden perusahaan diminta menjelaskan mengenai biodata personal yang berkaitan dengan institusi dimana responden bekerja, diantaranya nama lengkap, nama perusahaan, alamat email resmi berdomain perusahaan, website perusahaan, jabatan pekerjaan, negara lokasi perusahaan berjalan, sektor utama perusahaan, ukuran perusahaan (kuantitas staff), kategori lulusan yang diterima sebagai staff (internship, sarjana-magister (S1/S2), MBA), dan yang terakhir lokasi negara yang paling banyak kandidat diterima sebagai staff perusahaan [11].

2. Analisa perguruan tinggi bisnis terbaik

Poin penilaian ini, responden perusahaan diminta menjelaskan mengenai pendapat mereka terhadap perguruan tinggi yang mempunyai lulusan terbaik, baik dalam hal bisnis maupun keseluruhan, diantaranya adalah memilih 10 perguruan tinggi domestik sesuai tempat tinggal responden yang mempunyai lulusan terbaik, memilih 30 perguruan tinggi luar negeri yang mempunyai lulusan terbaik, memberi tahu daftar perguruan tinggi dari staff perusahaan yang bekerja dalam perusahaan tersebut, serta bidang disiplin ilmu yang paling banyak direkrut oleh perusahaan tersebut [11].

Selain itu mengidentifikasi juga sekolah bisnis dimana responden perusahaan diminta untuk menjelaskan beberapa hal, diantaranya adalah mengidentifikasi 10 sekolah bisnis yang alumninya berpotensi direkrut oleh perusahaan, perekrutan staff bergelar MBA (Master of Business Administration), membiayai staff pada program

executive MBA untuk mengembangkan potensi, menjelaskan pentingnya spesialisasi MBA dalam perusahaan, serta memilih 5 perguruan tinggi yang unggul dalam gelar MBA (ada 10 kategori spesialisasi, yang berarti ada 50 perguruan tinggi yang dipilih).

3. Kualifikasi untuk staff bergelar MBA

Poin penilaian ini, responden perusahaan diminta menjelaskan pendapatnya mengenai kualifikasi staff yang bergelar MBA, diantaranya adalah memilih region negara yang mempunyai sekolah bisnis terbaik dan menciptakan dampak tinggi untuk bisnis, lalu menjelaskan mengenai ketentuan yang ideal untuk staff bergelar MBA untuk bekerja (lama waktu kerja per hari, gaji, bonus, tunjangan), divisi/posisi kerja yang sesuai dengan staff bergelar MBA, rata-rata gaji yang ditawarkan oleh perusahaan untuk freshgraduate.

4. Pembahasan umum

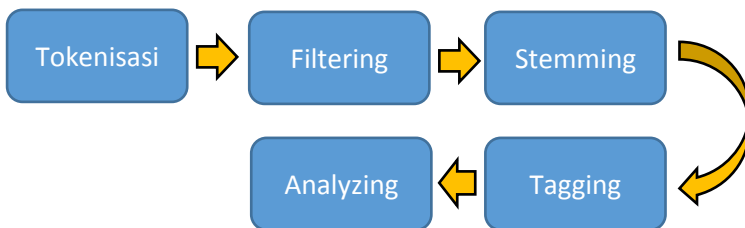
Poin penilaian ini, responden perusahaan diminta menjelaskan pendapatnya mengenai kebutuhan skill apa yang harus dimiliki oleh calon staff (bergelar MBA maupun reguler), diantaranya adalah skill yang dibutuhkan untuk freshgraduate, seperti pengalaman internasional, pengetahuan tentang subjek, kepemimpinan, pengalaman magang, kemampuan individu, dan kemampuan bahasa. Selain itu diminta menjelaskan mengenai skill yang dibutuhkan oleh calon staff MBA, seperti kemampuan individu, interaksi dengan lingkungan, kepemimpinan, prestasi akademik, pemikiran strategis, kemampuan bahasa, kewiraswastaan, keterampilan IT, manajemen resiko, pengalaman, pemasaran, keterampilan keuangan, komunikasi, tanggung jawab sosial, dan e-business [11].

Untuk kriteria ketiga hingga keenam, poin yang dijadikan penilaian adalah jumlah dari masing-masing. Kriteria ini tidak

mempunyai ketentuan poin khusus seperti kriteria sebelumnya yang bersifat kualitatif.

2.4 Teks Mining

Teks mining merupakan proses menggali data teks yang didapatkan dari sumber data berupa dokumen (word, pdf, kutipan, dan sejenisnya). Teks mining disebut juga dengan teknik mengekstraksi pola dari sebuah dokumen. Tujuan utama dari teks mining adalah untuk mendapatkan informasi yang berguna dari sekumpulan dokumen dengan melakukan pencarian kata-kata yang dapat mewakili isi dari sebuah dokumen yang kemudian dapat dianalisa. Kegunaan dari penggunaan teks mining adalah sebagai pengkategorian teks (*text categorization*) dan pengelompokan teks (*text clustering*) [12]. Proses umum dalam teks mining seperti pada Gambar 2.1.



Gambar 2.1 Proses Utama Teks Mining

2.5 Teknik Grabbing

Merupakan salah satu cara untuk mengambil konten teks atau object yang ditampilkan dalam situs website yang kemudian akan diletakkan dalam website milik pribadi. Secara umum teknik ini akan menduplikasi konten website tujuan untuk ditampilkan dalam website kita, akan tetapi dari sini banyak informasi dan kegunaan lain yang dapat dikembangkan. Salah satunya adalah sebagai salah satu bahan teks mining dimana dengan tujuan untuk menganalisa konten teks dalam website tersebut. Teknik grabbing dapat dikembangkan sendiri oleh developer sesuai dengan

kebutuhan yaitu kegunaan dan konten situs website sasaran, oleh karena itu teknik ini bisa dikatakan tidak fleksibel karena jika ingin melakukan grabbing pada situs website tertentu maka harus merubah pola konten yang akan diambil. Beberapa cara menggunakan teknik grabbing diantaranya adalah fungsi cURL, fungsi `file_get_contents`, library simple HTML dom parser, dan pemanfaatan ReGex (*Regular Expression*).

2.5.1 Fungsi cURL

Merupakan sebuah fungsi yang berguna untuk mengambil konten dari sebuah situs website. cURL dapat digunakan untuk melakukan simulasi request method POST ke server dan juga menggunakan COOKIE. Keuntungan menggunakan cURL adalah beberapa situs website yang melindungi kontennya dengan menggunakan cookie dan post method, dengan menggunakan cURL dapat mengakses konten yang terproteksi tersebut. Beberapa fungsi yang diakomodasi oleh cURL seperti terlihat pada Kode Sumber 1:

1. `curl_init()`, untuk inialisasi curl library.
2. `curl_setopt()`, untuk mengatur pilihan dari curl.
3. `curl_exec()`, untuk mengeksekusi/menjalankan query.
4. `curl_close()`, untuk menghentikan curl system.

```
$options = array(
    CURLOPT_CUSTOMREQUEST => "GET",
    CURLOPT_POST           => false,
    CURLOPT_FOLLOWLOCATION => true,
    CURLOPT_CONNECTTIMEOUT => 120,
    CURLOPT_TIMEOUT       => 120
);

$ch = curl_init( $url );
curl_setopt_array( $ch, $options );
$content = curl_exec( $ch );
curl_close( $ch );
$header['content'] = $content;
```

Kode Sumber 1 Syntax cURL dengan setting option url

2.5.2 Fungsi File_get_contents

Merupakan cara untuk membaca isi file ke dalam string. Karena akan menggunakan teknik pemetaan memori, jika hal ini didukung oleh server, untuk meningkatkan kinerja. Isi file dalam konteks grabbing adalah, bagian konten situs website yang akan diterjemahkan menjadi string, kemudian akan dianalisa sesuai kebutuhan dengan mengambil informasi dari tag konten bagian tertentu menggunakan ReGex (*Regular Expression*). Kode dapat dilihat pada Kode Sumber 2.

```
file_get_contents(path,include_path,context,start,max_length)
```

Kode Sumber 2 Syntax Fungsi file_get_contents

2.5.3 Simple HTML Dom Parser

Merupakan sebuah syntax dalam Bahasa pemrograman PHP yang berfungsi untuk menguraikan elemen-elemen HTML untuk diambil dan dirubah atau dihapus pada tag/elemen-elemen tertentu menjadi konten baru yang diinginkan. Dalam PHP sendiri sudah disediakan library HTML DOM yang dapat diunduh terpisah dengan banyak keuntungan. Kode terlihat pada Kode Sumber 3.

```
<?php
    require_once 'simple_html_dom.php';
    $html = file_get_html("http://hieppies.blogspot.com");
    $html = file_get_html("index.html");
    $html = file_get_html("/menu/index.html");
    $html = str_get_html("<html>
        <body>
            <p>Hi, Cantik!.</p>
            <p>Lagi ngapain?.</p>
        </body>
    </html>");
?>
```

Kode Sumber 3 Syntax Penggunaan Library Simple HTML Dom Parser

2.5.4 Regular Expression (ReGex)

Merupakan teknik yang digunakan untuk pencocokan string teks, seperti karakter tertentu, kata-kata, atau pola karakter. ReGex memiliki 2 fungsi utama yaitu untuk mencari dan mengganti, mencari suatu pola tertentu dalam text lalu menggantinya menjadi pola yang lain. Dalam penggunaannya untuk grabbing situs website, ReGex biasanya dikombinasikan dengan fungsi `file_get_contents` dari PHP. Mengambil konten menggunakan fungsi `file_get_contents` sedangkan menganalisa hasil dan mencari string konten yang dibutuhkan menggunakan regular expression.

2.6 Pusat Data *Scopus*

Scopus merupakan database bibliografi yang meng-index abstrak dan sitasi/kutipan untuk artikel jurnal akademik. Scopus merupakan salah satu produk bibliograsi dari Elsevier. Scopus muncul sejak tahun 1995, kurang lebih 21 tahun Scopus meng-index jurnal dari para akademisi di seluruh dunia. Hingga saat ini kurang lebih terdapat 22.000 judul artikel jurnal dan lebih dari 5.000 penerbit [12].

Scopus meng-index artikel jurnal akademik dari berbagai bidang ilmu. Maka dari itu banyak publikasi pemeringkatan perguruan tinggi lain seperti Scimago menggunakan jasa dari Scopus sebagai pusat database jurnal akademik. Scopus sendiri sangat berhati-hati dan selektif untuk meng-index artikel jurnal akademik. Tidak semua artikel jurnal di database lain seperti Google Scholar, ataupun Web of Science dapat ter-index oleh Scopus. Scopus hanya meng-index berdasarkan kualitas artikel jurnal tersebut.

Scopus juga menyediakan layanan untuk developer berupa Scopus API (Application Program Interfaces) sebagai sarana untuk mengembangkan informasi berdasarkan database Scopus, hanya saja untuk menggunakan layanan ini hanya untuk institusi yang berlangganan (*Subscribe*) saja [13]. Tidak semua orang dan institusi dapat menggunakan layanan ini. Keuntungan

dari database Scopus ini adalah mempunyai atribut data dan informasi yang lengkap, diantaranya nama author, jumlah publikasi, jumlah sitasi, ORCID, dan grafik perkembangan keaktifan author [13]. Selain itu dukungan layanan API sangat lengkap dan juga kontennya dapat dipercaya.

2.7 Google Scholar

Google Scholar merupakan layanan database metadata publikasi yang dibuat oleh Google. Dalam Google Scholar memungkinkan pengguna melakukan pencarian materi berupa teks dalam berbagai format publikasi. Index Google Scholar mencakup jurnal online dan publikasi ilmiah sehingga semua jurnal online bisa tercover dalam Google Scholar [14]. Tujuan utama dari Google Scholar adalah menyusun paper seperti yang dilakukan peneliti, dengan memperhatikan kelengkapan teks, nama penulis yang menampilkan artikel, serta frekuensi penggunaan kutipan artikel dalam literatur akademis lainnya.

Dalam Google Scholar, seorang peneliti dapat mendaftarkan akun dirinya dalam sistem sebagai biodata untuk setiap paper yang dipublikasikan. Dalam setiap biodata penulis terdapat beberapa indikator yang cukup unik sebagai bahan acuan bahwa paper yang dipublikasikan oleh penulis tersebut berkualitas dan terpercaya atau tidak [15]. Hal ini dapat dilihat dari jumlah paper yang dipublikasikan penulis yang terindex oleh Google Scholar, jumlah sitasi (*citation*), nilai h-index, dan nilai i10-index [16]. Berikut penjelasan masing-masing indikator dalam Google Scholar:

1. Jumlah Publikasi

Merupakan jumlah paper yang dipublikasikan oleh penulis yang terindex oleh Google Scholar. Disini dapat diketahui juga tahun publikasi dan jumlah sitasi/kutipan terhadap paper tersebut. Semakin banyak orang lain yang mengutip paper tersebut akan semakin tinggi peringkat posisi paper dalam daftar publikasi penulis tersebut. Jumlah publikasi ini dihitung sejak penulis aktif dalam Google Scholar.

2. Jumlah Sitasi

Merupakan jumlah total kutipan dari orang lain yang mengutip paper penulis tersebut. Semakin banyak orang yang mengutip paper penulis maka akan semakin banyak pula total sitasinya. Terdapat dua versi dalam jumlah sitasi pada Google Scholar, terdapat jumlah total (*all citation*) yang merupakan jumlah keseluruhan sejak penulis aktif dalam Google scholar, serta jumlah sejak 5 tahun terakhir (Since 2012) yang dapat dilihat bahwa kualitas sitasi dari paper tersebut baik atau tidak sejak 5 tahun terakhir.

3. Nilai H-Index

Merupakan ukuran index yang diusulkan oleh seorang fisikawan di Universitas California bernama Jorge E Hirsch pada tahun 1985. H-Index ini masih terus diteliti keakuratannya. H-Index sendiri adalah nilai index yang didapatkan dari jumlah paper sebanyak h paper penulis dengan jumlah kutipan untuk setiap *paper* minimal sama dengan nilai h tersebut [16].

4. Nilai i10-Index

Merupakan nilai index yang didapatkan dari jumlah 2 *paper* dari penulis yang dikutip minimal oleh 10 *paper* dari orang lain. Sehingga kondisi minimal dari kualitas *paper* tersebut dapat diketahui oleh cara ini.

2.8 *Science and Technology Index (SINTA)*

Science and Tecnology Index atau biasa disebut dengan SINTA merupakan pusat index, sitasi dan kepakaran terbesar di Indonesia. Sistem informasi riset berbasis website ini menawarkan akses cepat, mudah dan komprehensif untuk mengukur unjuk kerja peneliti, institusi dan jurnal di Indonesia. Sinta menyediakan benchmark dan analisis, identifikasi kekuatan riset tiap institusi, memperlihatkan kolaborasi penelitianm menganalisis tren penelitian dan direktori pakar. SINTA memberikan informasi mengenai tolok ukur institusi, kolaborasi, analisis penelitian serta direktori akademisi di Indonesia [17].

Beberapa informasi yang dapat digali dari database online SINTA [18], antara lain:

1. Jurnal yang diterbitkan oleh institusi pendidikan dan penelitian tinggi yang disajikan dalam bentuk peringkat.
2. Institusi pendidikan tinggi yang mempunyai kontribusi dalam penelitian, berisikan informasi jumlah author akademisi serta total sitasi yang dihasilkan dari masing-masing institusi pendidikan dan penelitian tinggi yang disajikan dalam bentuk peringkat.
3. Author akademisi yang mempunyai kontribusi dalam penelitian, berisikan informasi jumlah sitasi yang dimiliki, skor h-index, dan i10-index.
4. Detail informasi setiap institusi pendidikan tinggi:
 - a. Jumlah sitasi Google Scholar.
 - b. Jumlah dokumen Google Scholar.
 - c. Jumlah artikel pada Scopus database.
 - d. Jumlah non artikel pada Scopus database.
 - e. Jumlah sitasi dari Scopus.
 - f. Jumlah akademisi.
 - g. Daftar akademisi beserta kualitas sitasi (h-index, i10-index) dari Google Scholar dan Scopus database.
 - h. Jumlah jurnal artikel, book chapters, dan conference paper dari akademisi maupun institusi.
 - i. Skor penilaian menurut SINTA.

2.9 Pangkalan Data Pendidikan Tinggi (PDDIKTI)

Pangkalan Data Pendidikan Tinggi atau lebih umum disebut PDDIKTI merupakan pusat kumpulan data penyelenggara pendidikan tinggi seluruh Indonesia. Kementerian Riset, Teknologi, dan Pendidikan Tinggi merupakan kementerian dalam pemerintahan Indonesia yang berfokus pada penyelenggaraan urusan bidang riset, teknologi dan pendidikan tinggi [19].

Data dalam PDDIKTI merupakan hasil sinkronisasi yang dikelola oleh masing-masing perguruan tinggi di Indonesia, sehingga akan lebih update dalam penyebaran informasi, serta

terintegrasi secara nasional. Pangkalan data ini menjadi salah satu instrumen pelaksanaan penjaminan mutu. Data yang terangkum dalam PDDIKTI harus mudah diakses oleh stakeholder (lembaga akreditasi, pemerintah, masyarakat), sehingga menerapkan prinsip transparansi perguruan tinggi.

Beberapa data yang disediakan oleh PDDIKTI dalam sistem yang dapat dilihat [19], diantaranya:

1. Jumlah perguruan tinggi di Indonesia.
2. Jumlah program studi setiap perguruan tinggi.
3. Jumlah mahasiswa (jenis kelamin, kelompok bidang).
4. Jumlah dosen (jenis kelamin, ikatan kerja/status kepegawaian, pendidikan terakhir, jabatan akademik).

2.10 Alexa Traffic Rank

Alexa Traffic Rank merupakan sistem yang menyediakan fasilitas informasi tentang peringkat *webhost*, yang dilihat dari jumlah *traffic* pengunjung yang masuk kedalam situs *webhost* tersebut. Dengan kata lain Alexa Traffic Rank merupakan salah satu *tools* untuk *Search Engine Optimization* (SEO) website yaitu sitem untuk menganalisa dan mempengaruhi tingkat keterlihatan (*visibilitas*) sebuah situs pada mesin pencari [20].

Alexa Traffic Rank mempunyai nilai peringkat dari nilai terkecil yaitu niali angka 1 (satu) hingga seterusnya (berjuta-juta) yang mana dalam perhitungan sistem di Alexa Traffic Rank apabila semakin kecil nilai Alexa Traffic Rank, nilai kepopuleran situs akan semakin bagus. Selain itu Alexa Traffict Rank juga menyediakan API untuk pengembangan (*developer*) perangkat lunak berbasis analisa SEO yang tidak berbayar (*free*).

2.11 MOZ Analytics

MOZ Analytics merupakan salah satu *tools* yang digunakan untuk mengetahui jumlah pencarian *webhost*. MOZ Analytics dikembangkan oleh SEOMoz yang berbasis di Seattle, Washington, Amerika Serikat. Salah satu product dari MOZ yang berfokus pada pemeringkatan dan analisa adalah MOZ Rank. MOZ Rank mengkuantifikasi kepopuleran link *webhost*. Halaman

webhost mendapatkan skor MOZ Rank berdasarkan halaman lain yang terhubung dengan halaman *webhost* tersebut (ditautkan). Semakin banyak tautan dari halaman *webhost*, maka akan semakin tinggi skor dari MOZ Rank. Dengan kata lain MOZ Analytics dengan berbagai macam produk turunannya merupakan sebuah tools untuk *Search Engine Optimization* (SEO). MOZ Analytics mengembangkan API untuk *developer* SEO yang tidak berbayar (*free*).

2.12 Bahasa Pemrograman Python

Python merupakan bahasa pemrograman interpretatif multiguna dengan filosofi perancangan yang berfokus pada tingkat keterbacaan kode. Python sendiri mulai diluncurkan di masyarakat sejak tahun 1991 oleh Guido van Rossum dengan nama vendor Python Software Foundation. Python diklaim sebagai bahasa pemrograman yang menggabungkan kapabilitas, kemampuan dengan sintaksis kode yang jelas dan dilengkapi dengan pustaka (*library*) yang lengkap dan komprehensif.

Python didistribusikan dengan beberapa lisensi yang berbeda dari beberapa versi yang diluncurkan. Namun pada prinsipnya python dapat diperoleh dan dipergunakan secara bebas, bahkan untuk kepentingan komersial. Lisensi Python tidak bertentangan dengan definisi Open Source dan General Public License [21]. Python terdiri dari dua buah versi terbaru yaitu 3.4.2, dan 2.7.8 yang bebas digunakan oleh developer. Versi Python ini rilis pada tahun 2014. Kelebihan yang dimiliki oleh bahasa pemrograman Python sebagai berikut:

1. Memiliki kepustakaan (*library*) yang luas, dalam Python sudah disediakan modul-modul siap pakai sesuai kebutuhan developer.
2. Memiliki tata bahasa yang mudah dipelajari.
3. Memiliki aturan layout (*indentation*) yang mudah untuk dilakukan pengecekan.
4. Berfokus pada orientasi objek.
5. Memiliki sistem pengelolaan memori otomatis.

6. Modular, mudah dikembangkan dengan menciptakan modul-modul baru sesuai kebutuhan.

2.13 Library Python

Beberapa pustaka (*library*) yang digunakan dalam membangun sistem Tugas Akhir yang berhubungan dengan *teks mining*, diantaranya:

1. Natural Language ToolKit (NLTK)

NLTK merupakan pustaka utama untuk proses pengolahan bahasa alami (*natural language process*). Pustaka ini berperan penting dalam proses teks mining seperti klasifikasi, tagging, tokenisasi, dan pemrosesan semantic. Pustaka ini diimplementasikan pada bahasa pemrograman python, selain itu pustaka ini juga didukung oleh dokumentasi yang lengkap, mudah untuk dipahami, dan forum diskusi yang banyak membantu [22]. Beberapa pustaka NLTK yang diterapkan dalam Tugas Akhir ini, antara lain:

- a. NLTK corpus stopwords

Merupakan pustaka yang menyediakan corpus berisi keyword kata-kata stopwords, kata umum yang kurang penting keberadaannya didalam konteks dokumen. Corpus stopwords ini dikembangkan dalam berbagai bahasa yang disediakan oleh NLTK.

- b. NLTK Tokenize

Merupakan pustaka untuk melakukan preprocessing dokumen dengan melakukan tokenize dokumen menjadi sebuah kata tunggal.

- c. Porter Stemmer

Merupakan pustaka untuk proses stemming dalam sebuah dokumen, dengan menghapus imbuhan kata. Stemmer ini mengadopsi berbagai tipe bahasa utam yang ada didunia.

- d. Snowball Stemmer

Merupakan pustaka lain untuk proses stemming. Pustaka stemming yang dimiliki NLTK bervariasi

dengan tujuan pengembangan untuk mencapai hasil yang lebih bagus. Dari hasil penelitian sebelumnya, proses stemming dalam natural language process (NLP) menjadi isu utama yang harus terus dipecahkan. Snowball Stemmer ini merupakan salah satu pengembangan dari Porter Stemmer yang mempunyai keuntungan lebih.

e. WordNet

WordNet merupakan database leksikan untuk bahasa Inggris. Fungsi utama pustaka ini adalah untuk mengelompokkan sebuah kata kepada kata lain yang terhubung dalam rule sinonim atau biasa disebut synsets [24]. Penggunaan utama dari WordNet sebagai kombinasi antara kamus dan thesaurus.

f. Postag Word Tokenize

Merupakan pustaka untuk pemrosesan tagging *Part of Speech* (POS Tagger) kata dalam sebuah kalimat dalam dokumen. Pustaka ini berperan penting dalam penerapan perangkat lunak pada Tugas Akhir ini.

2. Skit Learn

Merupakan pustaka dalam bahasa python yang berfokus pada proses pembelajaran numeric, seperti algoritma klasifikasi, clustering, dan regresi. Pustaka ini dilengkapi dengan berbagai macam algoritma dan didesain berjalan dalam sistem vector Numpy dan SciPy [24].

Beberapa algoritma dari pustaka Skit Learn yang digunakan dalam Tugas Akhir ini, antara lain:

a. CountVectorizer

Pustaka ini ada pada modul `sklearn.feature_extraction` yang digunakan untuk mengekstrak fitur dalam format yang didukung oleh algoritma pembelajaran dari Skit Learn dari kumpulan data berformat teks maupun gambar. Fungsi utama CountVectorizer adalah mengkonversi kumpulan dokumen teks kedalam matriks token (*count term*).

- b. **TfidfTransformer**
Pustakan ini berfungsi sebagai transformasi matriks perhitungan menjadi representasi TF-IDF yang dinormalisasi. Merupakan skema pembobotan untuk mengurangi term yang sering muncul di corpus tertentu.
- 3. **Numpy**
Merupakan modul yang menyediakan objek perhitungan mathematic yang memudahkan dalam melakukan perhitungan. Objek utama yang disediakan adalah array yang dapat berperan sebagai matriks.
- 4. **TextBloob**
Textblob merupakan pustakan yang dapat melakukan berbagai proses terhadap data teks, seperti tokenisasi, analisa sentiment hingga proses translate bahasa kedalam beberapa bahasa yang umum didunia. Penggunaan textbloob dalam Tugas Akhir ini sebagai pustaka untuk melakukan uji coba proses translate dokumen berbahasa Indonesia menjadi bahasa Inggris sebelum dilakukan preprocessing [25].
- 5. **Option Parser**
Merupakan pustaka yang fleksibel untuk mengurai opsi dalam baris perintah pada command line terminal python. Keuntungan dari Option Parser adalah menggunakan gaya deklaratif bahasa yang lebih baik daripada modul yang sama sebelumnya. Option Parser atau Optparse memungkinkan user untuk menentukan opsi dalam sintaks GNU/POSIX dengan didukung oleh pilihan bantuan (*help*).
- 6. **MySQL Python dan MyQSL Client**
Merupakan modul dari MySQL untuk konektor bahasa pemrograman python kepada basis data MySQL. MySQL

Python dan MySQL Client digunakan untuk menyimpan hasil akhir teks mining dalam database.

7. Pustaka import tambahan

Selain menggunakan pustaka utama diatas yang membutuhkan konfigurasi lebih lanjut, ada beberapa pustaka default dari python, diantaranya:

a. String

Modul ini berisi beberapa konstanta dan kelas yang berfungsi sebagai metode pengolah string untuk tipe data string dalam pengolahan kata, seperti fungsi str, unicode, list, tuple, bytearray, buffer, xrange section.

b. Re (regular expression)

Modul ini menyediakan operasi pencocokan regular expression untuk pola tipe data string. Modul ini sesuai untuk pengolahan kata yang tanpa menggunakan library tambahan.

c. Os (Operating System)

Modul ini menyediakan cara portabel untuk menggunakan fungsi yang bergantung pada sistem operasi. Menjadi jalan untuk proses programming untuk memodifikasi fungsi pada sistem operasi.

d. Math

Modul ini menyediakan akses untuk mengeksplere fungsi matematika yang didefinisikan oleh standar bahasa C. Modul ini tidak dapat digunakan dengan bilangan kompleks.

e. Random

Modul ini mengimplementasikan proses generator bilangan acak untuk berbagai macam distribusi. Generator bilangan acak dapat ditentukan sesuai kebutuhan pengguna.

f. Time

Modul ini menyediakan berbagai fungsi yang berhubungan dengan waktu. Meskipun modul ini selalu tersedia, tidak semua fungsi tersedia di semua platform.

Sebagian besar fungsi yang didefinisikan dalam modul ini memanggil fungsi library bahasa C.

g. Csv

Format CSV (Comma Separated Values) merupakan format impor dan ekspor paling umum untuk spreadsheet dan database. Penggunaannya yang sederhana dan kurangnya standar untuk norma penulisan menjadi perbedaan disetiap platform yang menggunakan CSV.

2.14 PHP dan HTML

Hypertext Preprocessor atau yang biasa disebut dengan PHP merupakan bahasa script yang dapat disisipkan dalam kumpulan kode HTML [26]. PHP banyak dipakai dalam pemrograman website selain dengan HTML. PHP sendiri ditemukan oleh Rasmus Lerdorf pada tahun 1995, dan hingga sekarang PHP dikembangkan oleh vendor utama yaitu The PHP Group. Hingga saat ini PHP telah masuk dalam versi ke 7, atau biasa yang disebut dengan PHP 7. Kelebihan dari bahasa pemrograman PHP sebagai berikut:

1. Bahasa pemrograman yang tidak melakukan sebuah kompilasi dalam penggunaannya.
2. Web server yang mendukung PHP dapat ditemukan dimana-mana, diantaranya apache, IIS, Lighttpd, dan Xitami.
3. Dalam sisi pengembangan (*developer*) lebih mudah karena banyak organisasi maupun developer lain yang siap membantu mengembangkan variasi dari PHP sendiri.
4. Dalam sisi pemahaman, PHP adalah bahasa scripting yang paling mudah karena memiliki referensi yang baik dan banyak.
5. Open Source yang dapat digunakan di berbagai *Operating System* (OS) dan dapat dijalankan secara *runtime* melalui *console* serta dapat menjalankan perintah sistem.

2.15 Database Management System (DBMS) MySQL

MySQL merupakan sebuah perangkat lunak sistem manajemen basis (DBMS) data SQL. MySQL disebut database yang multithread, multi user dengan jumlah instalasi 6 juta kali di seluruh dunia yang membuat MySQL tersedia sebagai perangkat lunak gratis dibawah lisensi GNU General Public License (GPL). MySQL sendiri dikembangkan oleh Oracle pada tahun 1995. MySQL sendiri bisa dijalankan disemua tipe sistem operasi [27].

MySQL adalah sebuah implementasi dari sistem manajemen basis data relasional RDBMS yaitu sistem basis data yang entitas utamanya terdiri dari tabel-tabel yang mempunyai relasi dari satu tabel ke tabel yang lain. Setiap pengguna dapat secara bebas menggunakan MySQL, namun dengan batasan perangkat lunak tersebut tidak boleh dijadikan produk turunan yang bersifat komersial. MySQL sebenarnya merupakan turunan salah satu konsep utama dalam basis data yang telah ada sebelumnya, yaitu SQL (*Structured Query Language*). SQL adalah sebuah konsep pengoperasian basisdata, terutama untuk pemilihan atau seleksi dan pemasukan data, yang memungkinkan pengoperasian data dikerjakan dengan mudah secara otomatis.

2.16 Part-Of-Speech Tagger (POS Tagger)

Merupakan teknik yang dapat menerapkan penanda tag dari dokumen teks (corpus) dalam beberapa bahasa untuk menjadi kelas kata berdasarkan tipe struktur kalimat dalam dokumen. Struktur kata ini seperti kata benda (*noun*), kata kerja (*verb*), kata sifat (*adjective*) dan sebagainya. POS Tagger disebut juga grammatical tagging. Proses menandai sebuah kata dalam sebuah teks (corpus) yang sesuai dengan bagian pembicaraan tertentu. Berdasarkan definisi konteks, yaitu hubungan dengan kata-kata yang berdekatan yang terkait dalam frase, kalimat atau paragraf [28].

POS Tagging pada zaman dahulu dilakukan menggunakan cara manual yaitu dengan menggunakan jasa ahli bahasa untuk memetakan kelas dari kata dalam corpus. Untuk saat ini POS Tagging dilakukan dalam konteks komputasi linguistik,

menggunakan algoritma yang mengasosiasikan istilah diskrit, serta bagian pembicaraan yang tersembunyi (latent). Salah satu pengembangan POS Tagger yang populer saat ini adalah Stanford POS Tagger yang dibangun dalam bahasa Java. Struktur kelas kata POS Tagging digambarkan pada Tabel 2.1 [28].

Tabel 2.1 Struktur Kelas kata POS-Tagging

Tag	Deskripsi	Contoh Penerapan
CC	Coordin, conjunction	And, but, or
CD	Cardinal number	One, two
DT	Determiner	A, the
EX	Existential ‘there’	There
FW	Foreign word	Mea culpa
IN	Preposition/sub-conjunction	Of, in, by
JJ	Adjective	Yellow
JJR	Adj., comparative	Bigger
JJS	Adj., superlative	Wildest
LS	List item marker	1, 2, one
MD	Modal	Can, should
NN	Noun, sing. Or mass	Llama
NNS	Noun, plural	Llamas
NNP	Proper noun, sing.	IBM
NNPS	Proper noun, plural	Carolinas
PDT	Predeterminer	All, both
POS	Possessive ending	‘s
PRP	Personal pronoun	I, you, he
PRP\$	Possessive pronoun	Your, one’s
RB	Adverb	Quickly, never
RBR	Adverb, comparative	Faster
RBS	Adverb, superlative	Fastest
RP	Participle	Up, off
SYM	Symbol	_, %, &
TO	“to”	to
UH	Interjection	Ah, oops
VB	Verb base form	Eat

VBD	Verb past tense	Ate
VBG	Verb gerund	Eating
VBN	Verb past participle	Eaten
VBP	Verb on-3sg pres	Eat
VBZ	Verb 3-sg pres	Eats
WDT	Wh-determiner	Which, that
WP	Wh-pronoun	What, who
WP\$	Possessive wh-	Whose
WRB	Wh-adverd	How, where
\$	Dollar sign	\$
#	Pound sign	#
“	Left quote	‘ or “
”	Right quote	’ or ”
(Left parenthesis	[, (, {, <
)	Right parenthesis],), }, >
,	Comma	,
.	Sentence-final punc	. ! ?
:	Mid-sentence punc	: ; ... -- -

2.17 Bloom Taxonomy

Bloom Taxonomy merupakan sebuah cara untuk membagi cara belajar seseorang menjadi tiga ranah (*domain*) utama. Taksonomi ini dibuat untuk tujuan pendidikan. Dibuat pertama kali oleh Benjamin S. Bloom pada tahun 1956. Tiga ranah (*domain*) utama diantaranya [29]:

1. Cognitive Domain (ranah kognitif) berisi mengenai perilaku yang menekankan pada aspek intelektual, seperti pengetahuan, pengertian, dan keterampilan berpikir.
2. Affective Domain (ranah afektif) berisi mengenai perilaku yang menekankan pada aspek perasaan dan emosi, seperti minat, sikap, apresiasi, dan cara penyediaan diri.
3. Psychomotor Domain (ranah psychomotor) berisi mengenai aspek ketrampilan motorik seperti tulisan tangan, mengetik, dan mengoperasikan mesin.

Dari ketiga domain diatas, domain yang paling berpengaruh terhadap hasil intelektual adalah domain *cognitif*. Domain ini selanjutnya dibagi kedalam 6 level, diantaranya [29]:

1. Knowledge
Menunjukkan materi yang telah dipelajari sebelumnya dengan mengingat fakta, istilah, konsep dasar dan jawaban.
2. Comprehension
Menunjukkan pemahaman tentang fakta dan gagasan dengan mengatur, membandingkan, menerjemahkan, menafsirkan, memberikan deskripsi dan mengemukakan gagasan utama.
3. Application
Memecahkan masalah dengan menerapkan pengetahuan, fakta, teknik dan peraturan yang diperoleh dengan cara yang berbeda.
4. Analysis
Memeriksa dan memecahkan informasi menjadi beberapa bagian dengan mengidentifikasi motif atau sebab, serta membuat kesimpulan dan menemukan bukti untuk mendukung generalisasi.
5. Synthesis
mengompilasi informasi bersama-sama dengan cara yang berbeda dengan menggabungkan elemen dalam pola baru atau mengusulkan solusi alternatif.
6. Evaluation
Menyajikan dan mempertahankan pendapat dengan membuat penilaian tentang informasi, validitas gagasan atau kualitas kerja berdasarkan seperangkat kriteria.

2.18 *Helmholtz*

Prinsip *Helmholtz* pada konsep *text mining* menggunakan rumus yang sama seperti pada konsep *computer vision* yaitu NFA atau *Number of False Alarms*. Pada konsep *text mining*, NFA berpengaruh terhadap perhitungan ke-*meaningful*-an sebuah *keyword* [13]. Sebuah *keyword* dianggap *meaningful* jika nilai NFA-nya kurang dari 1 seperti pada persamaan (1) berikut.

$$NFA(w, P, D) = \binom{K}{m} \frac{1}{N^{m-1}} < 1 \quad (1)$$

Berikut ini merupakan *source code* metode *Helmholtz* yang melibatkan NFA dalam mengoptimasi pencarian *meaningful keyword*. [14]

Input: Store each document into an array from D_1 to D_N

```

Set corpus=[];
Add all the documents D1 to D2 into corpus Array;
L = length(corpus);
Set W=[];
for i := 0 to L do
    W = append(Uniquewords(corpus));
end for
K=[];
for i := 1 to length(W) do
    Set counter=0;
    for j := 1 to L do
        if W[i] == corpus[ j ] then
            counter=counter+1;
        end if
    end for
    K[j]=append(counter);
end for
B;
Window Size
x=[],y=[],z=[];
for i := 1 to N do
    1 Di
    for j := 1 to B do
        X[j]=appendDi[ j];
    if B 1 then
        for k = ( j + 1) to (B + 1) do
            y[k] = append(Di[k]);
            x=GetIntersection(x,y);
            j=j+1;
        end for
    end if
end for

```

```

        B=B+1;
    end for
end if
end for
end for
M = L/B;
for D(i = 1)toD(i = N) do
    m=[];
    for j = 1 to length(x) do
        counter =0;
        for k = 1 to l do
            if x[i] == Di[k] then
                counter=counter+1;
            end if
        end for
        m[ j] = append(counter);
    end for
end for
end for

```

Kode Sumber 2.4 Metode Helmholtz

2.19 *Labeled Latent Dirichlet Allocation (L-LDA)*

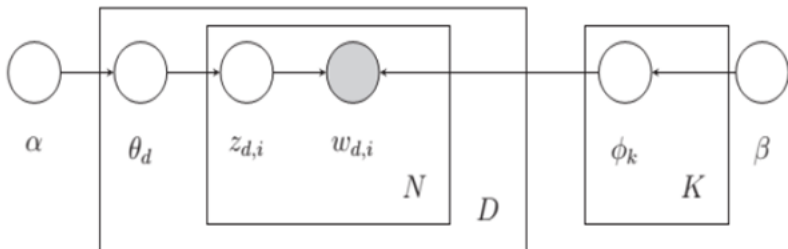
Metode *Labeled Latent Dirichlet Allocation* atau biasa disebut dengan metode L-LDA merupakan model umum probabilitas untuk multilabel corpora pada kumpulan dokumen. Metode L-LDA merupakan perkembangan dari metode *Latent Dirichlet Allocation* (LDA) yang digagas oleh David Blei pada tahun 2003 [30].

Metode LDA sendiri merupakan metode yang bersifat unsupervised, dimana tidak membutuhkan pembelajaran sebelumnya sebagai acuan kebenarannya. Metode LDA merupakan salah satu model untuk pemodelan topik [30]. Tujuan utama dari metode LDA ini adalah untuk menyimpulkan informasi utama yang tersembunyi dalam dokumen. Dalam model ini, terdapat 2 buah aspek bernilai penting dalam menyimpulkan informasi utama, yaitu aspek laten (topik/tersembunyi) dapat menjelaskan aspek observasi (teramati). Aspek observasi

merupakan dokumen yang diamati, sedangkan aspek laten merupakan topik yang ditentukan dari setiap kata pada dokumen yang diamati [31].

Tujuan dari penggunaan metode L-LDA adalah sebagai pengganti penggunaan kriteria yang bersifat kualitatif dalam pemeringkatan perguruan tinggi yaitu reputasi akademik. Reputasi akademik sebelumnya menggunakan metode survey yang cenderung dikerjakan secara manual [4], dengan metode ini akan diotomatisasi dengan memodelkan topik dari dataset abstrak paper akademisi perguruan tinggi. Abstrak tersebut termasuk dalam kategori/topik apa, yang nantinya setiap topik akan mempunyai bobot tersendiri, sehingga setiap topik dapat dikategorikan untuk menghasilkan scoring pada kriteria reputasi akademik.

Model generatif LDA dalam diagram model generatif [30] ditunjukkan pada Gambar 2.2.



Gambar 2.2 Diagram Representasi Model Generatif LDA

Berdasarkan diagram diatas didapat distribusi bersama (*joint-distribution*) yang dapat direpresentasikan dalam persamaan (2) [32]- [30].

$$p(w, z, \theta, \phi | \alpha, \beta) = p(\phi | \beta) p(\theta | \alpha) p(z | \theta) p(w | \phi_z) \quad (2)$$

Distribusi Dirichlet merupakan distribusi atas vektor, dimana elemen vektor merupakan bilangan diantara 0 dan 1,

dengan kata lain dapat dianggap sebagai bilangan probabilitas. Persamaan distribusi Dirichlet seperti pada persamaan (3) [32].

$$p(\theta|\alpha) = \frac{\Gamma(\sum_{i=1}^k \alpha)}{\prod_{i=1}^k \Gamma(\alpha)} \theta^{\alpha_1-1} \dots \theta^{\alpha_k-1} \quad (3)$$

Untuk menemukan aspek laten (topik/teramati) sebagai estimasi parameter metode LDA dapat ditemukan dengan metode estimasi Gibbs Sampling. Metode estimasi yang dapat digunakan dalam model LDA berdasarkan metode analisis informasi sampel dan informasi prior (distribusi posterior). Cara ini merupakan pendekatan simulasi untuk membangun distribusi bersama berdasarkan distribusi marginal, dalam proses estimasi parameter LDA [32].

Gibbs Sampling merupakan algoritma dari Markov Chain Monte Carlo (MCMC) dimana mempunyai konsep untuk menghitung joint probability distribution dengan cara melakukan sampling satu persatu terhadap setiap variabel berdasarkan nilai variabel lainnya. Gibbs Sampling ini memerlukan nilai peluang topik yang diasosiasikan untuk sebuah kata [33]. Persamaan Gibbs Sampling pada model LDA seperti pada persamaan (4).

$$P(z_i = t | z_{-i}, w_i,) \propto \frac{C_{-i,j}^{(w)} + \beta}{\sum_{w=1}^W C_{-i,t}^{(w)} + W\beta} \frac{C_{-i,t}^{(m_i)} + \alpha}{\sum_{m_i=1}^{m_i} C_{-i,t}^{(m_i)} + T\alpha} \quad (4)$$

Keterangan:

$C_{-i,j}^{(w)}$ = jumlah kata terhadap topik

$C_{-i,t}^{(m_i)}$ = jumlah dokumen terhadap topik

β = parameter penyeimbang jumlah kata terhadap topik, parameter ini diisikan bebas oleh user

α = parameter penyeimbang jumlah dokumen terhadap topik, parameter ini diisikan bebas oleh user

T = jumlah topik yang diinisiasi

W = jumlah kata yang diinisiasi

Berikut adalah proses generativ dari metode LDA [33] [30]:

Fase inisialisasi:

1. untuk setiap dokumen (dm) pada kumpulan dokumen (kd):
 - a. untuk setiap kata (w) dalam dokumen (dm):
 - i. membuat topik sample (z) secara random $\sim Mult(\frac{1}{K})$
 - ii. menambahkan $n_{-i,t}^{(w)}$, $n_{-i,t}^{(.)}$, $n_{-i,t}^{(dm_i)}$, dan $n_{-i}^{(dm_i)}$

Fase inferensi kesimpulan Gibbs Sampling:

1. untuk setiap iterasi:
 - a. untuk semua dokumen (dm) dalam kumpulan dokumen (kd):
 - i. untuk semua kata (w) dalam dokumen (dm):
 1. menghapus nilai topik sample (z) saat ini pada kata (w)
 2. mengurangi $n_{-i,t}^{(w)}$, $n_{-i,t}^{(.)}$, $n_{-i,t}^{(dm_i)}$, dan $n_{-i}^{(dm_i)}$
 3. untuk setiap topik (j), perulangan dari 0 hingga K-1:
 - a. menjumlahkan $p(z_i = t | z_{-i}, w_i)$
 4. membuat topik baru $z \sim p(z_i = t | z_{-i}, w_i)$
 5. memasukkan topik (z) pada kata (w)
 6. menambahkan $n_{-i,t}^{(w)}$, $n_{-i,t}^{(.)}$, $n_{-i,t}^{(dm_i)}$, dan $n_{-i}^{(dm_i)}$

Dalam proses generative LDA, proses komputasi distribusi posterior pada fase inferensi Gibbs Sampling pada langkah ke-4 menggunakan persamaan 3. Setelah estimasi inferensi Gibbs Sampling selesai, maka akan masuk dalam proses distribusi probabilitas topik pada kumpulan dokumen dengan menghitung nilai dari probabilitas topik terhadap dokumen $\vartheta^{(d_i)}$

dan probabilitas topik terhadap kata φ_j^w seperti pada persamaan (5) dan (6) [33].

$$\vartheta^{(d_i)} = \frac{n_{-i,j}^{(d_i)} + \alpha}{n_{-i}^{(d_i)} + T\alpha} \quad (5)$$

$$\varphi_j^w = \frac{n_{-i,j}^{(w)} + \beta}{n_{-i,j}^{(\cdot)} + W\beta} \quad (6)$$

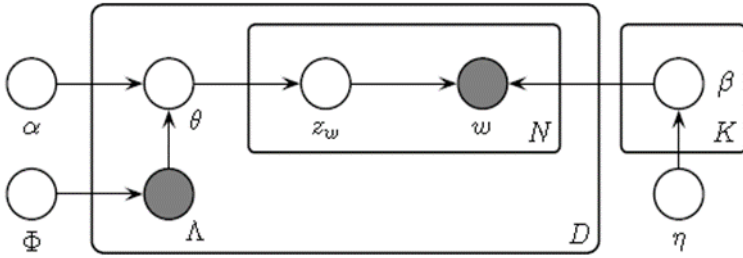
Keterangan:

$\vartheta^{(d_i)}$ = probabilitas dokumen terhadap topik

φ_j^w = probabilitas kata terhadap topik

Dalam tugas akhir ini menerapkan metode L-LDA sebagai metode pemodelan topik utama. Metode L-LDA sendiri merupakan model generatif untuk multilabel corpora, dimana menekankan pada penyediaan label sebelum melakukan komputasi [6]. Tujuan utama dari metode L-LDA sama seperti model LDA, yaitu sama-sama menyimpulkan informasi utama yang tersembunyi dalam dokumen.

Dalam model L-LDA terlebih dahulu membuat dokumen pelabelan untuk menghimpun topik yang akan digunakan sebagai langkah pengelompokan/klasifikasi. Setiap label berhubungan langsung dengan satu topik, sehingga setiap topik digambarkan sebagai satu label [6]. Setiap dokumen dalam metode L-LDA sebagai campuran topik yang mendasari serta topik yang dihasilkan pasti sesuai dalam lingkup label topik yang ditentukan [6]. Untuk representasi model generative *Labeled Latent Dirichlet Allocation* (L-LDA) dijelaskan pada Gambar 2.3.



Gambar 2.3 Diagram Model Generatif LDA

Konsep metode L-LDA adalah setiap dokumen direpresenyasikan dari kumpulan list kata dan list indikator binary topik (0, 1). Berikut adalah proses generative dari metode L-LDA [6]:

Fase inisialisasi topik:

1. untuk setiap topik (k), $k \in \{1, \dots, K\}$:
 - a. menghasilkan kosakata (vocabulary) dengan topik (k) $\sim \text{Dirichlet}(\cdot | \eta)$
 - $\beta_k = (\beta_{k,1}, \dots, \beta_{k,V})^T \sim \text{Dirichlet}(\cdot | \eta)$

Fase inferensi kesimpulan Gibbs Sampling:

1. untuk setiap dokumen (dm):
 - i. untuk setiap topik (k), $k \in \{1, \dots, K\}$:
 - i. generate label indikator binary topik $A^{(dm)} \in \{0, 1\} \sim \text{Ber}(\cdot | \Phi_k)$
 - b. generate dimensi vektor yang lebih rendah untuk memproyeksikan parameter vektor dari distribusi Dirichlet $\alpha^{(dm)} = L^{(dm)} x \alpha$

- c. menghasilkan distribusi multinomial campuran $\theta^{(dm)} = (\theta_{l_1}, \dots, \theta_{l_{M_{dm}}})^T \sim \text{Dir}(\cdot | \alpha^{(dm)})$
- d. untuk setiap iterasi sejumlah panjang kata dalam dokumen $\{1, \dots, N_d\}$:
- i. menghasilkan topik baru $z_i \in \{\lambda_1^{(d)}, \dots, \lambda_{M_d}^{(d)}\} \sim \text{Mult}(\cdot | \theta^{(d)})$
 - ii. menghasilkan list kata baru $w_i \in \{1, \dots, V\} \sim \text{Mult}(\cdot | \beta_{z_i})$

Persamaan Gibbs Sampling yang digunakan pada L-LDA mengacu pada (Griffiths dan Steyvers, 2004) pada persamaan (7).

$$P(z_i = j | z_{-i}) \propto \frac{n_{-i,j}^{w_i} + \eta_{w_i}}{n_{-i,j}^{(\cdot)} + \eta^T 1} x \frac{n_{-i,j}^{(dm)} + \alpha_j}{n_{-i,j}^{(dm)} + \alpha^T 1} \quad (7)$$

2.20 Term Frequency – Invers Document Frequency (TF-IDF)

Term Frequency – Inver Document Frequency atau biasa disebut dengan TF-IDF merupakan sebuah metode untuk mendapatkan bobot sebuah term dari sebuah corpus. Metode ini terkenal efisien, mudah dan memiliki hasil yang akurat. TF-IDF didapatkan dengan mencari nilai Term Frequency (TF) dan nilai Invers Documen Frequency (IDF) yang terlebih dahulu mencari nilai Document Fequency (DF) [37].

2.20.1 Term Frequency (TF)

Merupakan jumlah kemunculan suatu term dalam dokumen. Nilai TF kemudian dilogaritmikkan untuk mengurangi besarnya bilangan. Persamaan mencari TF seperti persamaan (8).

$$Tf = tfdt \quad (8)$$

2.20.2 Invers Dokumen Frequency (IDF)

Sebelum mencari IDF terlebih dahulu mencari nilai DF yaitu jumlah dokumen dimana sebuah term muncul. Setelah mendapatkan nilai DF kemudian menghitung nilai IDF yang berfungsi mengurangi bobot suatu term jika kemunculannya banyak tersebar di seluruh koleksi dokumen [37]. Persamaan DF seperti pada persamaan (9), dan IDF pada persamaan (10).

$$Df = dft \quad (9)$$

$$Idf_t = \log_{10}\left(\frac{N}{df_t}\right) \quad (10)$$

Keterangan:

- Tf = term frequency
- Df = dokumen frequency
- IDF = invers dokumen frequency
- N = jumlah dokumen

2.21 Akreditasi Perguruan Tinggi

Akreditasi perguruan tinggi merupakan salah satu cara untuk menilai kualitas perguruan tinggi secara keseluruhan, yang bertujuan untuk mengetahui komitmen perguruan tinggi terhadap kapasitas dan efektifitas pendidikan yang diselenggarakan, yang didasarkan pada standar akreditasi yang telah ditetapkan oleh badan akreditasi nasional [38].

Di Indonesia pihak yang berwenang untuk melakukan akreditasi institusi perguruan tinggi adalah Badan Akreditasi Nasional Perguruan Tinggi (BAN-PT). Akreditasi ini dilakukan oleh tim asesor yang turun langsung kepada perguruan tinggi terkait untuk dinilai dari beberapa hal. Berikut adalah poin-poin penting dalam penilaian akreditasi institusi oleh BAN-PT [38].

1. Menilai kesesuaian visi, misi, tujuan dan sasaran, serta strategi pencapaian dalam perguruan tinggi dengan kondisi yang sudah ada sekarang.
2. Menilai tata pamong (kepengurusan), kepemimpinan, sistem pengelolaan, dan penjaminan mutu.

3. Menilai kualitas mahasiswa dan lulusan.
4. Menilai sumber daya manusia (tenaga pengajar dan administrasi).
5. Menilai kesesuaian kurikulum, model pembelajaran, serta suasana akademik.
6. Menilai kebutuhan pembiayaan, sarana dan prasarana, serta sistem informasi dalam perguruan tinggi.
7. Menilai tentang penelitian, pelayanan/pengabdian masyarakat, serta kerjasama.

Penilaian akreditasi ini dilakukan dengan cara survey langsung terhadap perguruan tinggi, sehingga data yang didapatkan dapat objektif terhadap perguruan tinggi tersebut. Akreditasi yang dilakukan oleh BAN-PT tersebut mempunyai 2 jenis, yaitu akreditasi institusi dan akreditasi profesi.

Akreditasi institusi yang merupakan penilaian kualitas institusi dengan memperhatikan poin-poin penilaian diatas. Sedangkan akreditasi profesi merupakan penilaian kualitas terhadap bidang studi/departemen/jurusan dalam perguruan tinggi tersebut, yang mana akan menentukan kualitas untuk akreditasi institusi juga. Berikut adalah score penilaian akreditasi BAN-PT.

- a) Nilai 601 – 700, mendapatkan peringkat A
- b) Nilai 501 – 600, mendapatkan peringkat B
- c) Nilai 401 – 500, mendapatkan peringkat C
- d) Nilai 0 – 400, mendapatkan peringkat N/A (not applicable)

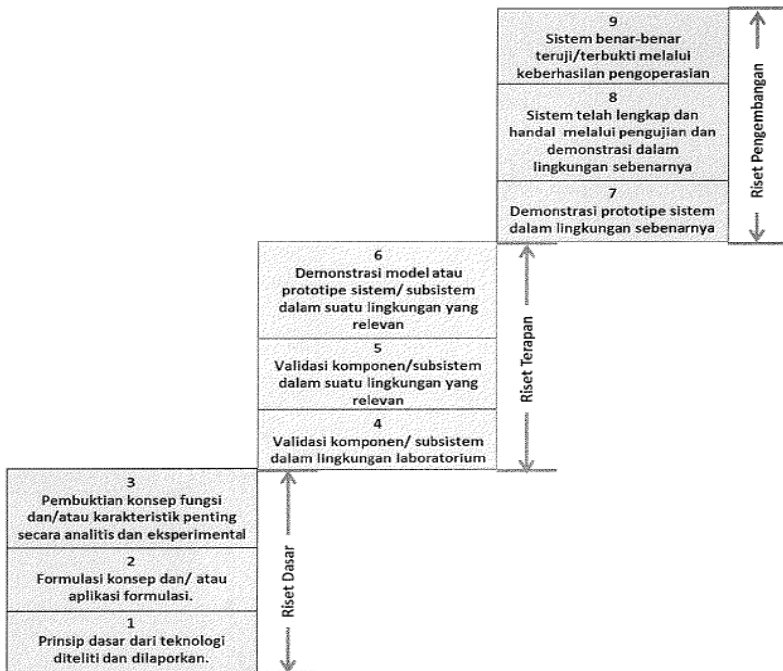
Dalam Tugas Akhir ini, salah satu kriteria penilaian yang menjadi acuan adalah kriteria reputasi employer. Dengan menggunakan kriteria akreditasi perguruan tinggi oleh BAN-PT yang dijelaskan pada poin penilaian kualitas mahasiswa dan lulusan sudah mengandung kriteria reputasi employer yang dibutuhkan.

2.22 Tingkat Kesiapterapan Teknologi (TKT)

Tingkat Kesiapterapan Teknologi atau yang biasa disebut dengan TKT merupakan tingkat kondisi kematangan atau kesiapterapan suatu hasil penelitian (*research*) dan pengembangan prototype tertentu yang selanjutnya akan masuk

pada tahap komersialisasi, yang diukur secara sistematis dengan tujuan untuk dapat dimanfaatkan oleh pengguna, baik oleh pemerintah, maupun masyarakat luas [38].

Setiap jenis penelitian (*research*) dan pengembangan prototype dikelompokkan menjadi 9 tingkatan dengan masing-masing memiliki indikator yang berbeda-beda. Seperti pada Gambar 2.4.



Gambar 2.4 Hirarki Tingkat Kesiapterapan Teknologi (TKT)

Berdasarkan hirarki tingkatan diatas, dapat dikelompokkan menjadi 3 kelompok tingkatan besar [38], di antaranya:

1. Riset dasar untuk TKT tingkat 1-3

Dalam kelompok ini, penelitian yang dimaksud adalah penelitian yang berfokus pada pembahasan metode, pengembangan teori, pembahasan gagasan produk. Penelitian ini akan bertujuan untuk membuat produk yang bermanfaat bagi masyarakat.

2. Riset terapan untuk TKT tingkat 4-6

Dalam kelompok ini, penelitian yang dimaksud adalah penelitian yang berfokus pada yang menghasilkan produk, demonstrasi model atau prototype subsistem dalam lingkungan yang relevan, dan berfokus pada validasi dalam lingkungan laboratorium. Penelitian ini sudah membuat produk/prototype.

3. Riset pengembangan untuk TKT tingkat 7-9

Dalam kelompok ini, penelitian yang dimaksud adalah penelitian yang berfokus pada pengembangan produk yang sudah ada sehingga membuat riset pendukung baru untuk produk yang sudah ada.

BAB III

ANALISIS DAN PERANCANGAN SISTEM

Bab ini membahas tahap analisis permasalahan dan perancangan Tugas Akhir. Analisis permasalahan membahas permasalahan yang diangkat dalam pengerjaan Tugas Akhir. Solusi yang ditawarkan oleh penulis juga dicantumkan pada tahap permasalahan analisis ini. Analisis kebutuhan mencantumkan kebutuhan-kebutuhan yang diperlukan perangkat lunak. Selanjutnya dibahas mengenai perancangan sistem yang dibuat. Perancangan direpresentasikan dengan diagram UML (*Unified Modelling Language*).

3.1. Tahap Analisis

Tahap analisis mendefinisikan kebutuhan yang akan dipenuhi dalam pembangunan aplikasi pemeringkatan universitas.

3.1.1 Deskripsi Umum

Pada tugas akhir ini dibangun suatu aplikasi untuk melakukan pemeringkatan universitas. Aplikasi dapat digunakan untuk menunjang pemanfaatan peringkat universitas di bidang pendidikan. Data yang digunakan adalah data abstrak *paper* masing-masing universitas. Data keluaran dari aplikasi merupakan peringkat universitas dari yang tertinggi hingga terendah. Data keluaran tersebut akan digunakan sebagai bahan analisis terkait sistem maupun peringkat universitas.

3.1.2 Arsitektur Perangkat Lunak

Seluruh proses yang terdapat pada aplikasi pemeringkatan universitas dan metode *Helmholtz* dijalankan pada komputer. Aplikasi menerima data masukan berupa abstrak 50 *paper* masing-masing 16 universitas yang telah dipilih. Aplikasi pemeringkatan universitas dibangun menggunakan bahasa pemrograman HTML. Sedangkan khusus untuk metode *Helmholtz* digunakan bahasa pemrograman Python. Aplikasi ini akan berjalan pada *platform* desktop dengan sistem operasi Windows.

Setelah menerima data masukan, aplikasi mengolah data masukan dan menampilkan data keluaran. Data keluaran adalah urutan peringkat universitas dari yang tertinggi hingga terendah.

3.1.3 Analisis Aktor

Aktor yang terdapat pada aplikasi pemeringkatan universitas ini adalah pengunjung situs pemeringkatan universitas. Pengguna atau pengunjung dapat memasukkan skor persentase masing-masing kriteria penilaian pemeringkatan universitas.

3.1.4 Analisis Permasalahan

Permasalahan utama yang diangkat dalam pembuatan Tugas Akhir ini adalah bagaimana mengimplementasikan suatu aplikasi berbasis web untuk memberikan peringkat pada universitas di Indonesia. Dengan penggunaan metode tertentu, sistem diharapkan dapat dengan akurat memberikan peringkat kepada universitas di Indonesia.

3.1.5 Spesifikasi Kebutuhan Perangkat Lunak

Spesifikasi kebutuhan dalam sistem ini mencakup kebutuhan fungsional. Kebutuhan fungsional berisikan proses-proses yang dibutuhkan dalam sistem dan harus dijalankan. Kebutuhan fungsional sistem dideskripsikan dalam Tabel 3.1

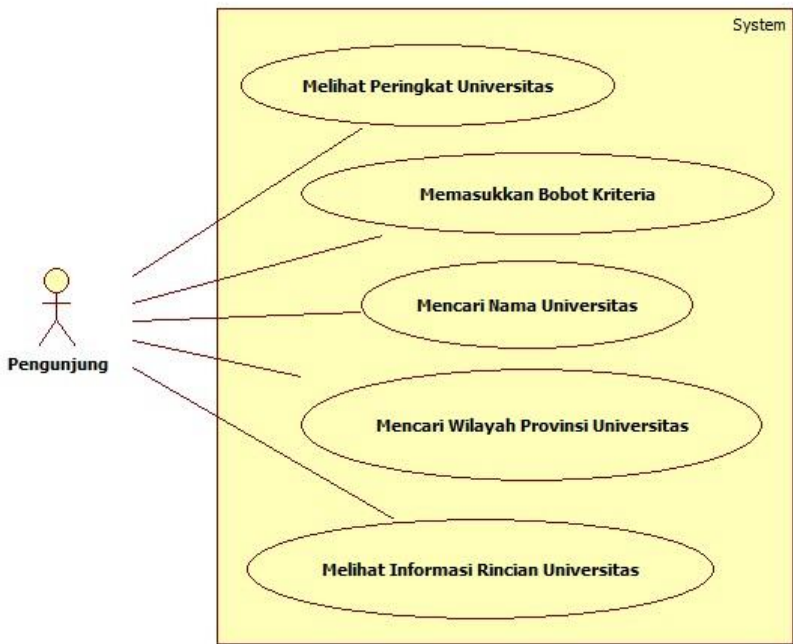
Tabel 3.1 Daftar Kebutuhan Fungsional Sistem

Kode Kebutuhan	Kebutuhan Fungsional	Deskripsi
F-001	Melihat Peringkat Universitas	Pengunjung dapat melihat peringkat berdasarkan bobot. Sistem akan menampilkan peringkat universitas secaraurut beserta masing-masing nilai yang telah dikali

		persentase per kriteria.
F-002	Memasukkan Bobot Kriteria	Pengunjung dapat memasukkan bobot sendiri. Sistem akan menampilkan universitas sesuai masukkan pengunjung.
F-003	Mencari Nama Universitas	Pengunjung dapat disediakan kolom <i>search</i> untuk mencari universitas berdasarkan namanya masing-masing. Sistem akan menampilkan universitas yang dicari beserta masing-masing nilai persentase kriteria.
F-004	Mencari Wilayah Provinsi Universitas	Pengunjung dapat mencari universitas berdasarkan masing-masing wilayahnya. Sistem menyediakan sebuah peta agar dapat dipilih oleh pengguna dalam melakukan pencarian.
F-005	Melihat Rincian Informasi Universitas	Pengguna dapat melihat informasi rincian masing-masing universitas setelah dilakukan proses pemeringkatan oleh sistem.

3.1.6 Kasus Penggunaan

Kasus penggunaan yang dibutuhkan pada sistem sesuai dengan analisa yang telah dilakukan. Diagram kasus penggunaan dapat dilihat pada Gambar 3.1.6.1 dan kode kasus penggunaan ada pada tabel 3.2.



Gambar 3.1 Diagram Kasus Penggunaan

Tabel 3.2 Keterangan Kode Kasus Penggunaan

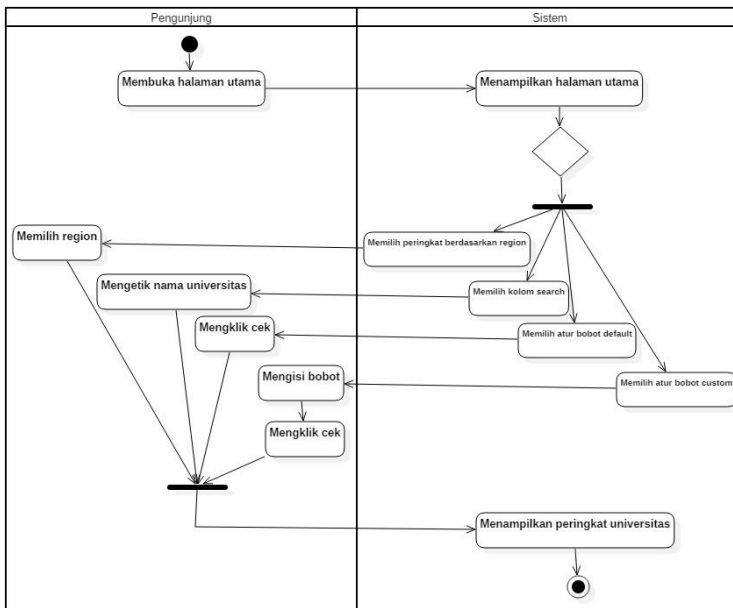
Kode Kasus Penggunaan	Kasus Penggunaan
UC-001	Melihat Peringkat Universitas
UC-002	Memasukkan Bobot Kriteria
UC-003	Mencari Nama Universitas
UC-004	Mencari Wilayah Provinsi Universitas
UC-005	Melihat Rincian Informasi Universitas

3.1.6.1 UC-001 Melihat Peringkat Universitas

Tabel 3.3 Spesifikasi Kasus Melihat Peringkat Universitas

Nama	Melihat Peringkat Universitas
Nomor	UC-001
Description	Kasus penggunaan ini digunakan untuk melihat peringkat universitas di Indonesia yang telah dihitung
Tipe	Fungsional
Aktor	Pengunjung
Kondisi Awal	Pengunjung sudah masuk ke sistem
Kondisi Akhir	Sistem menampilkan peringkat universitas di Indonesia
Alur Normal	<ol style="list-style-type: none"> 1. Pengunjung membuka halaman utama situs pemeringkatan universitas 2. Sistem menampilkan halaman utama situs pemeringkatan universitas 3. Pengunjung memilih salah satu sub pilihan yang terdapat pada halaman utama situs <ol style="list-style-type: none"> a. Sub pilihan kolom <i>search</i> b. Sub pilihan Peringkat Berdasarkan <i>Region</i> c. Sub pilihan Atur Bobot Pemeringkatan <i>tab default</i> d. Sub pilihan Atur Bobot Pemeringkatan <i>tab custom</i> 4. Sistem menampilkan halaman peringkat universitas di Indonesia 5. Kasus Penggunaan Berakhir
Alur Alternatif	<ol style="list-style-type: none"> A1. Pengunjung memilih sub pilihan kolom <i>search</i> <ol style="list-style-type: none"> 1. Pengunjung mengisi nama universitas pada kolom <i>search</i> 2. Berlanjut ke alur normal langkah 4 A2. Pengunjung memilih sub pilihan Peringkat Berdasarkan <i>Region</i> <ol style="list-style-type: none"> 1. Pengunjung memilih satu <i>region</i> dari peta yang diperlihatkan 2. Berlanjut ke alur normal langkah 4 A3. Pengunjung memilih sub pilihan Atur Bobot Pemeringkatan <i>tab default</i> <ol style="list-style-type: none"> 1. Pengunjung klik cek di salah satu kolom pemeringkatan universitas

	<p>2. Berlanjut ke alur normal langkah 4</p> <p>A4. Pengunjung memilih sub pilihan atur bobot pemeringkatan <i>tab custom</i></p> <p>1. Pengunjung mengisi bobot di beberapa baris di salah satu kolom pemeringkatan universitas</p> <p>2. Pengunjung klik cek di salah satu kolom pemeringkatan universitas</p> <p>3. Berlanjut ke alur normal langkah 4</p>
--	---



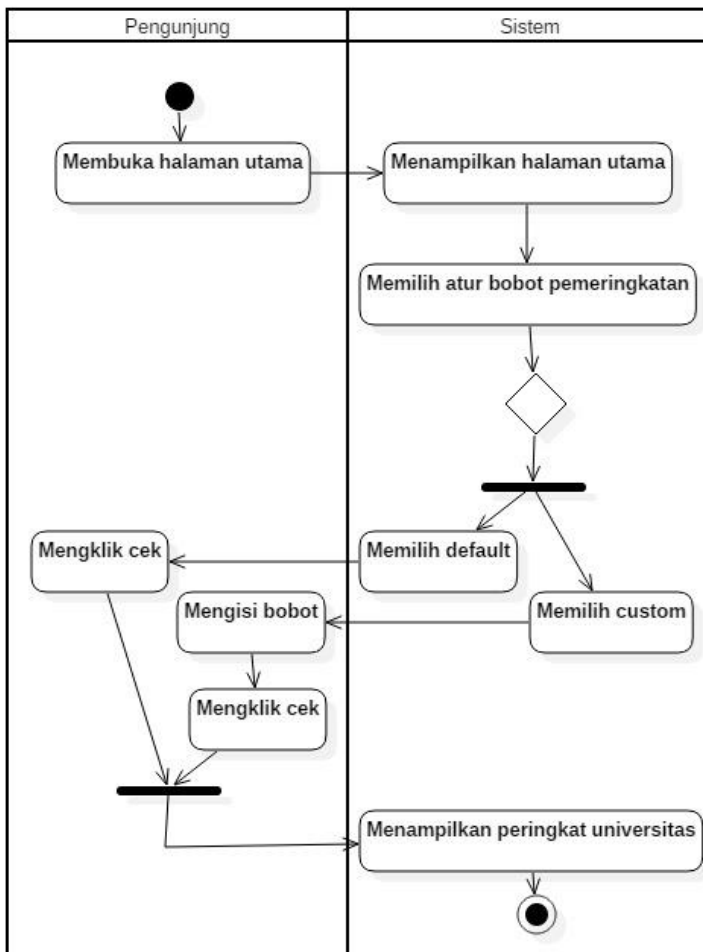
Gambar 3.2 Diagram Aktivitas UC-001

3.1.6.2 UC-002 Memasukkan Bobot Kriteria

Tabel 3.4 Spesifikasi Kasus Memasukkan Bobot Kriteria

Nama	Memasukkan Bobot Kriteria
Nomor	UC-002
Description	Kasus penggunaan ini digunakan untuk memasukkan

	bobot kriteria penilaian pemeringkatan universitas
Tipe	Fungsional
Aktor	Pengunjung
Kondisi Awal	Pengunjung sudah masuk ke sistem
Kondisi Akhir	Sistem menampilkan daftar peringkat universitas sesuai nilai persentase yang telah dimasukkan
Alur Normal	<ol style="list-style-type: none"> 1. Pengunjung membuka halaman utama situs pemeringkatan universitas 2. Sistem menampilkan halaman utama situs pemeringkatan universitas 3. Pengunjung memilih sub pilihan Atur Bobot Pemeringkatan 4. Pengunjung memilih salah satu <i>tab</i> pada sub pilihan Atur Bobot Pemeringkatan <ol style="list-style-type: none"> a. <i>Tab default</i> b. <i>Tab custom</i> 5. Sistem menampilkan halaman utama peringkat universitas di Indonesia 6. Kasus penggunaan berakhir
Alur Alternatif	<p>A1. Pengunjung memilih <i>tab default</i></p> <ol style="list-style-type: none"> 1. Pengunjung klik cek di salah satu kolom pemeringkatan universitas 2. Berlanjut ke alur normal langkah 4 <p>A2. Pengunjung memilih <i>tab custom</i></p> <ol style="list-style-type: none"> 1. Pengunjung mengisi bobot di beberapa baris di salah kolom pemeringkatan universitas 2. Pengunjung klik cek di salah satu kolom pemeringkatan universitas 3. Berlanjut ke alur normal langkah 4

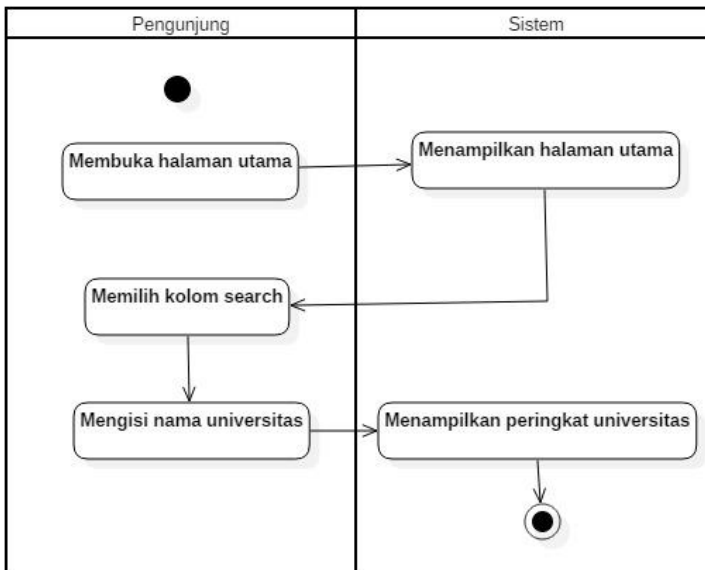


Gambar 3.3 Diagram Aktivitas UC-002

UC-003 Mencari Nama Universitas

Tabel 3.5 Spesifikasi Kasus Penggunaan Mencari Nama Universitas

Nama	Mencari Nama Universitas
Nomor	UC-003
Description	Kasus penggunaan ini digunakan untuk mencari universitas berdasarkan namanya
Tipe	Fungsional
Aktor	Pengunjung
Kondisi Awal	Pengunjung sudah masuk ke sistem
Kondisi Akhir	Sistem menampilkan peringkat universitas di Indonesia sesuai nama
Alur Normal	<ol style="list-style-type: none"> 1. Pengunjung membuka halaman utama situs pemeringkatan universitas 2. Sistem menampilkan halaman utama situs pemeringkatan 3. Pengunjung memilih sub pilihan kolom <i>search</i> 4. Pengunjung mengisi kolom <i>search</i> 5. Sistem menampilkan daftar universitas berdasarkan nama yang dicari 6. Kasus penggunaan berakhir
Alur Alternatif	



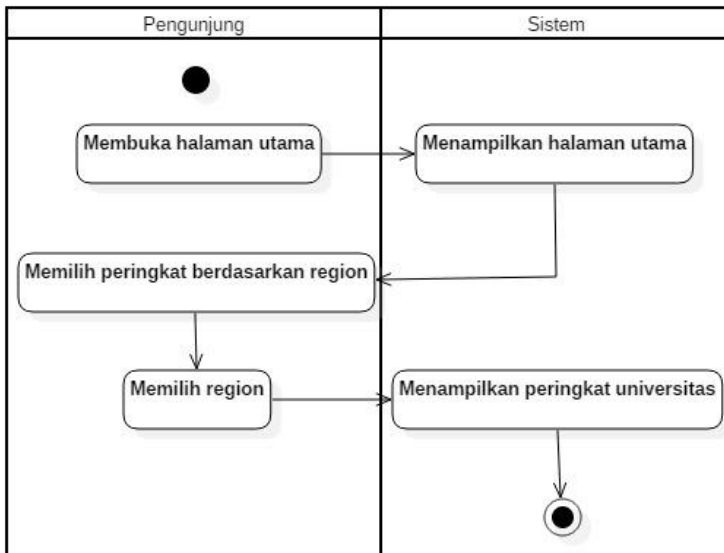
Gambar 3.4 Diagram Aktivitas UC-003

3.1.6.3 UC-004 Mencari Wilayah Provinsi Universitas

Tabel 3.6 Spesifikasi Kasus Penggunaan Mencari Wilayah Provinsi Universitas

Nama	Mencari Wilayah Provinsi Universitas
Nomor	UC-004
Description	Kasus penggunaan ini digunakan mencari peringkat universitas yang telah dimasukkan berdasarkan provinsi
Tipe	Fungsional
Aktor	Pengunjung
Kondisi Awal	Pengunjung sudah masuk ke sistem
Kondisi Akhir	Sistem menampilkan peringkat universitas berdasarkan wilayah provinsi yang telah dipilih

Alur Normal	<ol style="list-style-type: none"> 1. Pengunjung membuka halaman utama situs pemeringkatan universitas 2. Sistem menampilkan halaman utama situs pemeringkatan universitas 3. Pengunjung memilih sub pilihan Peringkat Berdasarkan Region 4. Pengunjung memilih wilayah provinsi yang diinginkan 5. Sistem menampilkan halaman wilayah provinsi yang telah dipilih 6. Kasus penggunaan berakhir
Alur Alternatif	



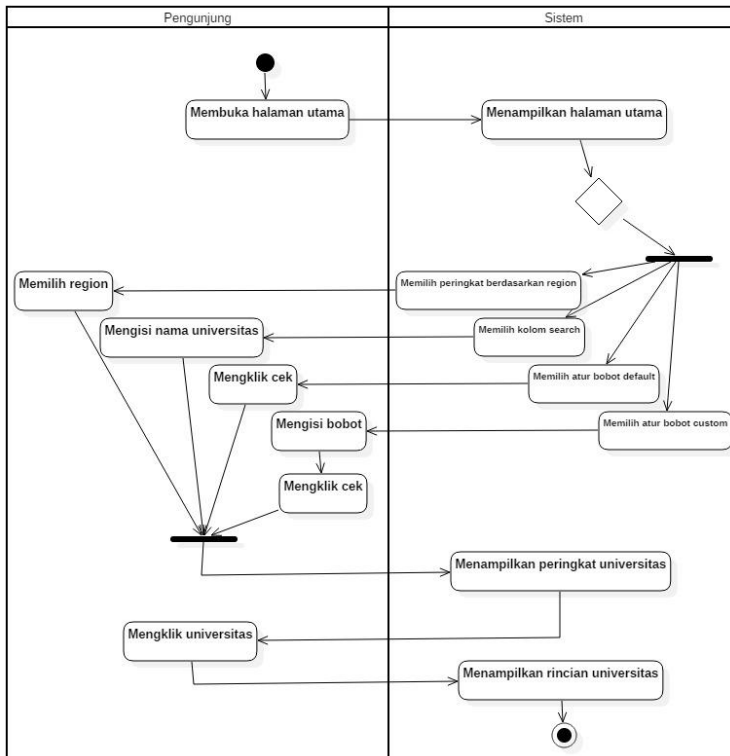
Gambar 3.5 Diagram Aktivitas UC-004

3.1.6.4 UC-005 Melihat Rincian Informasi Universitas

Tabel 3.7 Spesifikasi Kasus Penggunaan Melihat Rincian Informasi Universitas

Nama	Melihat Rincian Informasi Universitas
Nomor	UC-005
Description	Kasus penggunaan ini digunakan untuk melihat rincian informasi peringkat universitas yang dipilih secara lengkap
Tipe	Fungsional
Aktor	Pengunjung
Kondisi Awal	Pengunjung sudah masuk ke sistem
Kondisi Akhir	Sistem menampilkan detail informasi universitas
Alur Normal	<ol style="list-style-type: none"> 1. Pengunjung membuka halaman utama situs pemeringkatan universitas 2. Sistem menampilkan halaman utama situs pemeringkatan universitas 3. Pengunjung memilih salah satu sub pilihan yang terdapat pada halaman utama situs <ol style="list-style-type: none"> a. Sub pilihan kolom <i>search</i> b. Sub pilihan Peringkat Berdasarkan <i>Region</i> c. Sub pilihan Atur Bobot Pemeringkatan tab <i>default</i> d. Sub pilihan Atur Bobot Pemeringkatan tab <i>custom</i> 4. Sistem menampilkan peringkat universitas 5. Pengguna mengklik universitas yang diinginkan 6. Sistem menampilkan rincian universitas 7. Kasus penggunaan berakhir
Alur Alternatif	<ol style="list-style-type: none"> A1. Pengunjung memilih sub pilihan kolom <i>search</i> <ol style="list-style-type: none"> 1. Pengunjung mengisi nama universitas pada kolom <i>search</i> 2. Berlanjut ke alur normal langkah 4 A2. Pengunjung memilih sub pilihan Peringkat Berdasarkan <i>Region</i> <ol style="list-style-type: none"> 1. Pengunjung memilih satu region dari peta yang diperlihatkan 2. Berlanjut ke alur normal langkah 4

	<p>A3. Pengunjung memilih sub pilihan Atur Bobot Pemeringkatan tab <i>default</i></p> <ol style="list-style-type: none"> 1. Pengunjung klik cek di salah satu kolom pemeringkatan universitas 2. Berlanjut ke alur normal langkah 4 <p>A4. Pengunjung memilih sub pilihan atur bobot pemeringkatan tab <i>custom</i></p> <ol style="list-style-type: none"> 1. Pengunjung mengisi bobot di beberapa baris di salah satu kolom pemeringkatan universitas 2. Pengunjung klik cek di salah satu kolom pemeringkatan universitas 3. Berlanjut ke alur normal langkah 4
--	---



Gambar 3.6 Diagram Aktivitas UC-005

3.1.7 Label Kelas dalam Teks Mining Metode L-LDA

Penggunaan metode L-LDA yang bersifat supervised mengharuskan adanya keberadaan kelas atau bisa disebut juga dengan label. Label ini digunakan sebagai penanda pengetahuan dokumen masuk kedalam kondisi seperti apa. Kondisi dalam studi kasus ini adalah hirarki dari Tingkat Kesiapterapan Teknologi yang berjumlah 9 level. Berikut adalah daftar kelas yang digunakan dalam Tugas Akhir ini.

Tabel 3.8 Penjelasan Setiap Kelas Label

No.	Hirarki TKT	Kelas/Label	Keterangan
1	Level 1	T1	Prinsip dasar dari teknologi diteliti dan dilaporkan.
2	Level 2	T2	Formulasi konsep dan/atau aplikasi formulasi.
3	Level 3	T3	Pembuktian konsep fungsi dan/atau karakteristik penting secara analisa dan eksperimental
4	Level 4	T4	Validasi komponen atau subsistem dalam lingkup laboratorium
5	Level 5	T5	Validasi komponen atau subsistem dalam lingkungan yang relevan
6	Level 6	T6	Demonstrasi model atau prototype sistem atau subsistem dalam lingkungan yang relevan
7	Level 7	T7	Demonstrasi model dalam lingkungan yang sebenarnya
8	Level 8	T8	Sistem telah lengkap dan handal melalui pengujian dan demonstrasi dalam lingkungan sebenarnya
9	Level 9	T9	Sistem benar-benar teruji

			melalui keberhasilan pengoperasian
--	--	--	---------------------------------------

3.1.8 Ground Truth Pengujian Sistem

3.1.8.1 Peringkat QS World University Rankings

Dalam Tugas Akhir ini pemeringkatan keseluruhan dengan menggunakan 6 indikator penilaian mengacu pada kebenaran peringkat QS World University Rankings tahun 2016-2017 sesuai dengan waktu pengerjaan Tugas Akhir ini dimulai. Selain itu melakukan uji coba perbandingan jika mengacu pada Ground Truth peringkat QS World University Rankings dari beberapa tahun yang berbeda. Berikut adalah beberapa peringkat QS World University Rankings dan peringkat reputasi akademik dari berbagai jangka waktu.

1. QS 2015-2016

Peringkat universitas wilayah Indonesia berdasarkan QS World University Rankings 2015-2016.

Tabel 3.9 Peringkat Reputasi Akademik QS 2015-2016

Peringkat Reputasi Akademik	Nama Perguruan Tinggi
1	Institut Teknologi Bandung (ITB)
2	Universitas Indonesia (UI)
3	Universitas Gadjah Mada (UGM)
4	Universitas Muhammadiyah Surakarta (UMS)
5	Universitas Airlangga (UNAIR)
6	Institu Pertanian Bogor (IPB)
7	Universitas Diponegoro (UNDIP)
8	Universitas Brawijaya (UB)
9	Institut Teknologi Sepuluh Nopember (ITS)

10	Universitas Padjadjaran (UNPAD)
11	Universitas Islam Indonesia (UII)
12	Universitas 17 Agustus 1945 (UNTAG)
13	Universitas Bina Nusantara (BINUS)
14	Universitas Pelita Harapan (UPH)
15	Universitas Sebelas Maret (UNS)
16	Universitas Mataram (UNRAM)

Tabel 3.10 Peringkat QS World University Rankings 2015-2016

Peringkat QS	Nama Perguruan Tinggi
1	Universitas Indonesia (UI)
2	Institut Teknologi Bandung (ITB)
3	Universitas Gadjah Mada (UGM)
4	Universitas Airlangga (UNAIR)
5	Institu Pertanian Bogor (IPB)
6	Universitas Padjadjaran (UNPAD)
7	Universitas Diponegoro (UNDIP)
8	Universitas Muhammadiyah Surakarta (UMS)
9	Institut Teknologi Sepuluh Nopember (ITS)
10	Universitas Brawijaya (UB)
11	Universitas Bina Nusantara (BINUS)

2. QS 2016-2017

Peringkat universitas wilayah Indonesia berdasarkan QS World University Rankings 2016-2017.

Tabel 3.11 Peringkat Reputasi Akademik QS 2016-2017

Peringkat Reputasi Akademik	Nama Perguruan Tinggi
1	Institut Teknologi Bandung (ITB)
2	Universitas Indonesia (UI)
3	Universitas Gadjah Mada (UGM)
4	Universitas Airlangga (UNAIR)
5	Institu Pertanian Bogor (IPB)
6	Universitas Diponegoro (UNDIP)
7	Institut Teknologi Sepuluh Nopember (ITS)
8	Universitas Muhammadiyah Surakarta (UMS)
9	Universitas Brawijaya (UB)

Tabel 3.12 Peringkat QS World University Rankings 2016-2017

Peringkat QS	Nama Perguruan Tinggi
1	Universitas Indonesia (UI)
2	Institut Teknologi Bandung (ITB)
3	Universitas Gadjah Mada (UGM)
4	Universitas Airlangga (UNAIR)
5	Institu Pertanian Bogor (IPB)
6	Universitas Diponegoro (UNDIP)
7	Institut Teknologi Sepuluh Nopember (ITS)
8	Universitas Muhammadiyah Surakarta (UMS)
9	Universitas Brawijaya (UB)

3. QS 2017-2018

Peringkat universitas wilayah Indonesia berdasarkan QS World University Rankings 2017-2018.

Tabel 3.13 Peringkat Reputasi Akademik QS 2017-2018

Peringkat Reputasi Akademik	Nama Perguruan Tinggi
1	Institut Teknologi Bandung (ITB)
2	Universitas Gadjah Mada (UGM)
3	Universitas Indonesia (UI)
4	Universitas Airlangga (UNAIR)
5	Institu Pertanian Bogor (IPB)
6	Universitas Diponegoro (UNDIP)
7	Institut Teknologi Sepuluh Nopember (ITS)
8	Universitas Muhammadiyah Surakarta (UMS)
9	Universitas Brawijaya (UB)

Tabel 3.14 Peringkat QS World University Rankings 2017-2018

Peringkat QS	Nama Perguruan Tinggi
1	Universitas Indonesia (UI)
2	Institut Teknologi Bandung (ITB)
3	Universitas Gadjah Mada (UGM)
4	Universitas Airlangga (UNAIR)
5	Institu Pertanian Bogor (IPB)
6	Universitas Diponegoro (UNDIP)
7	Institut Teknologi Sepuluh Nopember (ITS)
8	Universitas Muhammadiyah Surakarta (UMS)
9	Universitas Brawijaya (UB)

3.1.8.2 Analisa Label Topik Dokumen

Penggunaan teks mining menggunakan metode *Labeled Latent Dirichlet Allocation* (L-LDA) membutuhkan acuan kebenaran sebagai perbandingan apakah keluaran sistem yang dibangun sudah sesuai dengan kebenaran dalam melakukan penentuan topik dalam indikator reputasi akademik.

Ground Truth yang digunakan adalah hasil analisa manual terhadap dokumen abstrak *paper* akademisi perguruan tinggi. Terdapat 800 dokumen sebagai dataset metode teks mining dari masing-masing perguruan tinggi, dan dianalisa manual untuk mencari label topik apa yang sesuai. Berikut adalah beberapa contoh hasil analisa manual terhadap dataset.

Tabel 3.15 Contoh Hasil Analisa Manual Sebagai Ground Truth Reputasi Akademik

No	Nama Perguruan Tinggi	No Dok	Nama Dokumen	Hasil Analisa
1	Institut Pertanian Bogor	1	IPB_001_01	T1, T2
		2	IPB_001_02	T1, T2
		3	IPB_001_03	T1, T2, T3
		4	IPB_05_10	T3, T4, T5
		5	IPB_05_14	T4, T5, T6, T7
2	Institut Teknologi Bandung	1	ITB_01_01	T4, T3, T5
		2	ITB_01_05	T4, T5
		3	ITB_01_22	T1, T2, T3
		4	ITB_01_09	T4, T5, T6, T7
		5	ITB_01_07	T4, T5

3.1.9 Dataset Pengujian Sistem

3.1.9.1 Data Dokumen Abstrak Akademisi Perguruan Tinggi

Dokumen abstrak paper akademisi didapatkan dari database metadata publikasi Google Scholar. Dalam Tugas Akhir

ini memfokuskan dokumen abstrak paper pada akademisi yang mempunyai bidang fokus teknik. Masing-masing perguruan tinggi diambil 50 dokumen abstrak *paper* teknik berdasarkan sitasi tertinggi dari akademisi yang terdaftar atas institusi perguruan tinggi di Google Scholar.

Dalam proses teks mining, 50 buah dataset abstrak paper perguruan tinggi digabung menjadi satu dokumen utuh yang memungkinkan adanya kata yang sama berjumlah lebih dari satu (duplikasi). Satu dokumen utuh ini selanjutnya akan menjadi masukan untuk proses tek mining tahap preprocessing. Penentuan dataset dokumen abstrak paper akademisi mempunyai rule seperti berikut:

1. Pencarian daftar akademisi dalam Google Scholar dengan keyword email yang terhubung dengan institusi atau keyword hosting institusi. Misal Institut Teknologi Sepuluh Nopember dicari menggunakan keyword “@its.ac.id” atau www.its.ac.id
2. Daftar akademisi yang diambil adalah yang mempunyai tingkat sitasi tertinggi dan berfokus pada bidang teknik.
3. Masing-masing dari akademisi yang mempunyai sitasi tertinggi diambil paper yang membahas teknik.
4. Dokumen abstrak paper difokuskan pada dokumen berbahasa Inggris.

Tahap proses teks mining dalam Tugas Akhir ini akan menghasilkan reduksi terhadap kata (*term*) yang ada di dalam dokumen abstrak *paper* akademisi. Reduksi terhadap kata (*term*) berperan penting untuk menyeleksi kata (*term*) yang tidak penting. Dilakukan analisa terhadap keberadaan kata (*term*) yang penting dan tidak penting dalam dataset untuk melihat rata-rata duplikasi term. Rincian jumlah kata pada dataset abstrak paper akademisi (16 perguruan tinggi teratas berdasarkan *Ground Truth*) seperti pada Tabel 1-16.

Tabel 3.16 Daftar Rincian Jumlah Kata Dataset Abstrak Paper
Perguruan Tinggi

No.	Perguruan Tinggi	Total Kata
1.	Institus Pertanian Bogor	10539
2.	Institut Teknologi Bandung	9287
3.	Institut Teknologi Sepuluh Nopember	9630
4.	Universitas 17 Agustus 1945	9146
5.	Universitas Airlangga	9457
6.	Universitas Bina Nusantara	7236
7.	Universitas Brawijaya	8651
8.	Universitas Diponegoro	8397
9.	Universitas Gadjah Mada	10214
10.	Universitas Indonesia	8823
11.	Universitas Islam Indonesia	7897
12.	Universitas Mataram	7527
13.	Universitas Muhammadiyah Surakarta	10330
14.	Universitas Padjadjaran	7774
15.	Universitas Pelita Harapan	7204
16.	Universitas Sebelas Maret	7807

3.1.9.2 Data Corpus Toxonomy Bloom

Corpus Taxonomy Bloom berfungsi sebagai dasar pengetahuan corpus tingkat kesiapterapan teknologi (TKT). Corpus Taxonomy Bloom mempunyai 6 level tingkatan yaitu Knowledge, Comprehension, Application, Analysis, Synthesis, dan Evaluation [29]. 6 level corpus Taxonomy Bloom akan digunakan untuk pengetahuan corpus TKT yang nantinya akan dikembangkan lagi menjadi 9 level (*class*). Dilihat dari jumlah keyword yang tercakup dalam masing-masing level Taxonomy Bloom dapat dikatakan kurang banyak dalam jumlah keyword jika dibandingkan dengan jumlah kata (*term*) pada dokumen dataset abstrak *paper* akademisi. Untuk selanjutnya akan diadakan proses breakdown corpus Taxonomy Bloom menjadi corpus TKT dengan melalui berbagai tahap. Rincian jumlah dari masing-masing corpus Taxonomy Bloom seperti pada Tabel 1-17.

Tabel 3.17 Daftar Rincian Jumlah Kata Corpus Taxonomy Bloom

No.	Level Taxonomy	Total Kata
1.	Knowledge	35
2.	Comprehension	29
3.	Application	36
4.	Analysis	51
5.	Synthesis	51
6.	Evaluation	46

3.1.9.3 Data Corpus Tingkat Kesiapterapan Teknologi (TKT)

Corpus tingkat kesiapterapan teknologi (TKT) merupakan corpus utama untuk proses pengenalan kata (*term*) dalam dokumen abstrak *paper* akademisi agar masuk dalam kategori level TKT yang mana. Corpus TKT ini berjumlah 9 level, dan didapatkan dari proses breakdown 6 level corpus Taxonomy Bloom.

Metode breakdown yang digunakan dalam penentuan corpus TKT ini adalah sebagai berikut:

1. Menggabungkan 6 level corpus Taxonomy Bloom menjadi satu dokumen besar.
2. Mengurutkan kata (*term*) dari corpus Taxonomy Bloom besar berdasarkan bobot konteks kata (*term*) dari yang paling dasar hingga tingkat penerapan. Tahap ini menggunakan kemampuan manusia (*expert*) untuk membedakan dan mensorting sesuai urutan.
3. Menghapus kata yang berjumlah lebih dari satu (duplikasi).
4. Memecah urutan konten kata (*term*) tersebut menjadi 9 bagian. Pemecahan ini didasarkan pada setiap urutan level TKT akan bergantung pada TKT selanjutnya. Sebagai contoh corpus TKT 2 pasti berkorelasi dengan TKT 1 dan TKT 3. Sehingga pemecahan ini mudah untuk dilakukan. Secara tidak langsung level TKT dapat diperoleh kata-kata dari berbagai macam kata pada level Taxonomy Bloom.

5. Dokumen hasil breakdown berjumlah 9 corpus kata TKT baru tanpa mengandung kata duplikasi.

Perbandingan dari 6 level corpus Taxonomy Bloom dengan 9 level corpus TKT seperti pada Tabel 1-18.

Tabel 3.18 Perbandingan Jumlah Kata Corpus TKT

Level Taxonomy Bloom	Jumlah Kata	Level TKT	Jumlah Kata
Knowledge	35	TKT 1	31
		TKT 2	24
Comprehension	29	TKT 3	32
Application	36	TKT 4	15
		TKT 5	32
Analysis	51	TKT 6	24
Synthesis	51	TKT 7	26
		TKT 8	30
Evaluation	46	TKT 9	34
Total	256	Total	256

3.2. Tahap Perancangan Sistem

Tahap ini meliputi perancangan basis data, perancangan tampilan antarmuka (*Interfaces*), perancangan alur proses penggunaan sistem antar pengguna, dan proses utama dalam proses teks mining fitur ekstraksi menggunakan metode *Helmholtz* dan topik modelling menggunakan metode *Labeled Latent Dirichlet Allocation* (L-LDA). Diharapkan dapat memenuhi tujuan dari pengembangan aplikasi ini.

3.2.1 Perancangan Basis Data

Perangkat lunak yang dibangun menggunakan basis data MySQL dengan bahasa syntax SQL. Terdapat 28 tabel untuk menampung hasil proses grabbing data online, hasil proses teks minning berupa skor reputasi akademik, hasil penampung skor

masing-masing kriteria, serta data perguruan tinggi. Terdapat 2 buah kegunaan utama dari basis data yang digunakan dalam pembuatan perangkat lunak, yaitu sebagai penyimpanan data final, dan penyimpanan data temporary update. Dengan rincian 16 tabel utama dan 12 tabel temporary sebagai penampung sementara data hasil Teknik grabbing.

3.2.1.1 Tabel Universitas

Identifikasi : universitas

Deskripsi isi : tabel untuk menyimpan data nama perguruan tinggi beserta informasi detail

<u>Id_field</u>	Deskripsi	Tipe & length	Null
<u>Id</u>	Variabel primary key	Int(2)	No
<u>Nama</u>	Variabel penampung nama	Varchar(255)	No
<u>Domain</u>	Variabel penampung domain webhost	Varchar(150)	No
<u>url_web</u>	Variabel penampung alamat link webhost	Varchar(150)	No
<u>Kota</u>	Variabel kota	Varchar(150)	No
<u>provinsi</u>	Variabel provinsi	Varchar(150)	No

3.2.1.2 Tabel Reputasi Akademik

Identifikasi : reputasi_akademik

Deskripsi isi : tabel untuk menyimpan nilai hasil proses teks mining metode L-LDA sebagai representasi nilai reputasi akademik

<u>Id_field</u>	Deskripsi	Tipe & length	Null
<u>Id</u>	Variabel primary key	Int(2)	No
<u>Nama_universitas</u>	Variabel berisi nama perguruan tinggi	Varchar(255)	No
<u>Skor_universitas</u>	Variabel penampung skor final reputasi akademik	Float	No
<u>Bobot_tota</u>	Variabel penampung jumlah	Int(2)	No

l	hasil reputasi akademik		
Jumlah_ dokumen	Variabel penampung jumlah dokumen olah	Int(2)	No
Tanggal_ akses	Varabel timestamp	Datetime	No

3.2.1.3 Tabel Akreditasi

Identifikasi : tampung_akreditasi

Deskripsi isi : tabel untuk menyimpan nilai hasil proses
grabbing akreditasi perguruan tinggi

Id_field	Deskripsi	Tipe & length	Null
Id	Variabel primary key	Int(2)	No
Nama_ universitas	Variabel berisi nama perguruan tinggi	Varchar(255)	No
Skor_ universitas	Variabel penampung skor final reputasi akademik	Float	No
Bobot_total	Variabel penampung jumlah hasil reputasi akademik	Int(2)	No
Jumlah_ dokumen	Variabel penampung jumlah dokumen olah	Int(2)	No
Tanggal_ akses	Varabel waktu akses	Datetime	No

3.2.1.4 Tabel Jumlah Mahasiswa

Identifikasi : jumlah_mahasiswa

Deskripsi isi : tabel untuk menyimpan jumlah mahasiswa yang
didapatkan dari data PDDIKTI

Id_field	Deskripsi	Tipe & length	Null
Id	Variabel primary key	Int(2)	No
Kode_ universitas	Variabel penampung kode status diakui universitas	Int(2)	No
Nama_uni versitas	Variabel berisi nama perguruan tinggi	Varchar(255)	No

Jumlah_tahun1_semester1	Variabel jumlah mahasiswa tahun1 pada semester1	Int(2)	No
Jumlah_tahun1_semester2	Variabel jumlah mahasiswa tahun1 pada semester2	Int(2)	No
Jumlah_tahun2_semester1	Variabel jumlah mahasiswa tahun2 pada semester1	Int(2)	No
Jumlah_tahun2_semester2	Variabel jumlah mahasiswa tahun2 pada semester2	Int(2)	No
Status_universitas	Variabel berisi keterangan status universitas	Varchar(5)	No

3.2.1.5 Tabel Jumlah Sitasi Sinta

Identifikasi : jumlah_sitasi_sinta

Deskripsi isi : tabel untuk menyimpan jumlah sitasi akademisi yang didapatkan dari database online SINTA

Id_field	Deskripsi	Tipe & length	Null
Id	Variabel primary key	Int(2)	No
Kode_universitas	Variabel penampung kode status diakui universitas	Int(2)	No
Nama_universitas	Variabel berisi nama perguruan tinggi	Varchar(250)	No
Jumlah_tahun1_semester1	Variabel jumlah mahasiswa tahun1 pada semester1	Int(2)	No
Jumlah_tahun1_semester2	Variabel jumlah mahasiswa tahun1 pada semester2	Int(2)	No
Jumlah_tahun2_semester1	Variabel jumlah mahasiswa tahun2 pada semester1	Int(2)	No
Jumlah_tahun2_semester2	Variabel jumlah mahasiswa tahun2 pada semester2	Int(2)	No

Status_uni veritas	Variabel berisi keterangan status universitas	Varchar(5)	No
-----------------------	--	------------	----

3.2.1.6 Tabel Jumlah Author Sinta

Identifikasi : jumlah_author_sinta

Deskripsi isi : tabel untuk menyimpan jumlah h-index dan i10-index akademisi yang didapatkan dari database online SINTA

Id_field	Deskripsi	Tipe & length	Null
Id	Variabel primary key	Int(2)	No
Id_data	Variabel tampung id data secara kolektif	Int(2)	No
Nomor	Variabel nomor akademisi dalam range satu kali iterasi	Int(2)	No
Nama_aut hor	Variabel penampung nama akademisi	Varchar(250)	No
Nama_uni veritas	Variabel penampung nama universitas	Varchar(250)	No
Jumlah_sit asi_scholar	Variabel jumlah sitasi	Int(2)	No
I10_index_ scholar	Vaiabel jumlah i10-index	Int(2)	No
H_index_s cholar	Variabel jumlah h_index	Int(2)	No
Waktu_aks es	Variabel penampung waktu akses	Tinytext	No

3.2.1.7 Tabel Presence

Identifikasi : presence

Deskripsi isi : tabel untuk menyimpan jumlah domain, subdomain dan repository webhost perguruan tinggi yang terindex Google

Id_field	Deskripsi	Tipe & length	Null
Id	Variabel primary key	Int(2)	No
url_web	Variabel penampung link	Varchar(255)	No

	webhost		
Domain_se arch	Variabel penampung link search webhost	Varchar(150)	No
Hasil_sear ch	Variabel penampung hasil search webhost	Int(2)	No
Tanggal_a kses	Variabel waktu akses	Datetime	No

3.2.1.8 Tabel Impact

Identifikasi : impact

Deskripsi isi : tabel untuk menyimpan jumlah page authority, domain authority dan backlink webhost perguruan tinggi

<u>Id_field</u>	Deskripsi	Tipe & length	Null
Id	Variabel primary key	Int(2)	No
url_web	Variabel penampung link webhost	Varchar(255)	No
Domain_w eb	Variabel penampung link webhost tanpa “www”	Varchar(150)	No
Page_auth ority	Variabel penampung nilai page authority	Int(2)	No
Domain_a uthority	Variabel penampung nilai domain authority	Int(2)	No
Backlink_e ksterna	Variabel penampung jumlah backlink	Int(2)	No
Tanggal_a kses	Variabel waktu akses	Datetime	No

3.2.1.9 Tabel Alexa Rank

Identifikasi : alexa_rank

Deskripsi isi : tabel untuk menyimpan peringkat webhost berdasarkan alexa rank

<u>Id_field</u>	Deskripsi	Tipe & length	Null
Id	Variabel primary key	Int(2)	No
url_web	Variabel penampung link	Varchar(255)	No

	webhost		
Rank_world	Variabel penampung peringkat webhost skala dunia	Int(2)	No
Rank_country	Variabel penampung peringkat webhost skala nasional	Int(2)	No
Country_name	Variabel penampung nilai domain authority	Varchar(150)	No
Tanggal_akses	Variabel waktu akses	Datetime	No

3.2.1.10 Tabel Kumpulan Skor Nilai

Identifikasi : kumpulan_skor_final

Deskripsi isi : tabel untuk menyimpan skor dari setiap kriteria pemeringkatan

Id_field	Deskripsi	Tipe & length	Null
Id	Variabel primary key	Int(2)	No
Kode_universitas	Variabel penampung kode status diakui universitas	Float	No
Nama_universitas	Variabel nama perguruan tinggi	Varchar(255)	No
Skor_reputasi_akademik_normalisasi	Variabel penampung skor reputasi akademik yang dinormalisasi	Float	No
Skor_nilai_akreditasi_normalisasi	Variabel penampung skor nilai akreditasi yang dinormalisasi	Float	No
Skor_jumlah_mahasiswa_normalisasi	Variabel penampung skor jumlah mahasiswa yang dinormalisasi	Float	No
Skor_jumlah_sitasi_normalisasi	Variabel penampung skor jumlah sitasi yang dinormalisasi	Float	No
Skor_jumlah_tenaga_pengajar_normalisasi	Variabel penampung skor jumlah tenaga pengajar yang dinormalisasi	Float	No

Skor_kualita s_sitasi_nor malisasi	Variabel penampung skor kualitas sitasi yang sudah dinormalisasi	Float	No
Skor_jumlah _phd_norma lisasi	Variabel penampung skor jumlah tenga pengajar S3 (Phd) yang sudah dinormlasiasi	Float	No
Skor_jumlah _paper_nor malisasi	Variabel penampung skor jumlah paper yang dinormalisasi	Float	No
Skor_jumlah jurnal_artike l	Variabel penampung skor jumlah jurna artikel yang dinormalisasi	Float	No
Skor_jumlah _paper_conf erence	Variabel penampung skor jumlah paper conference yang dinormalisasi	Float	No
Skor_precen se_normalisa si	Variabel penampung skor precense yang dinormalisasi	Float	No
Skor_impact _normalisasi	Variabel penampung skor impact yang dinormalisasi	Float	No
Skor_transpa rency_norma lisasi	Variabel penampung skor transparency yang dinormalisasi	Float	No
Skor_exellec en_normalis asi	Variabel penampung skor excellence yang dinormalisasi	Float	No
Skor_moz_a nalytic_nor malisasi	Variabel penampung skor moz analytic yang dinormalisasi	Float	No
Skor_alexa_ rank_normal isasi	Variabel penampung skor alexa rank yang dinormalisasi	Float	No
Skor_kualita s_dosen_nor malisasi	Variabel penampung skor kualitas dosen yang dinormalisasi	Float	No
Skor_kecuku pan_dosesn_ normalisasi	Variabel penampung skor kecukupan dosesn yang dinormalisasi	Float	No

Skor_kualita s_manajeme n_normalisa si	Variabel penampung skor kualitas manajemen yang dinormalisasi	Float	No
Skor_kualita s_kegiatan_k emahasiswaa n	Variabel penampung skor kualitas kegiatan mahasiswa yang dinormalisasi	Float	No
Sko_kualitas _kegiatan_p enelitian	Variabel penampung skor kualitas kegiatan penelitian yang dinormalisasi	Float	No

3.2.1.11 Tabel Peringkat 4ICU

Identifikasi : 4icu

Deskripsi isi : tabel untuk menyimpan skor dan peringkat
pemeringkatan 4ICU

Id_field	Deskripsi	Tipe & length	Null
Id	Variabel primary key	Int(2)	No
Nama_unive rsitas	Variabel penampung nama universitas	Varchar(255)	No
Skor_4icu	Variabel penampung skor 4ICU	Float	No
Peringkat_4i cu	Variabel penampung peringkat 4ICU	Int(2)	No

3.2.1.12 Tabel Peringkat DIKTI

Identifikasi : dikti

Deskripsi isi : tabel untuk menyimpan skor dan peringkat
pemeringkatan DIKTI Indonesia

Id_field	Deskripsi	Tipe & length	Null
Id	Variabel primary key	Int(2)	No
Nama_uni versitas	Variabel penampung nama universitas	Varchar(150)	No
Skor_dikti	Variabel penampung skor	Float	No

	DIKTI		
Peringkat_dikti	Variabel penampung peringkat DIKTI	Int(2)	No

3.2.1.13 Tabel Peringkat QS General

Identifikasi : qs_general

Deskripsi isi : tabel untuk menyimpan skor dan peringkat pemeringkatan QS rank versi general

<u>Id_field</u>	Deskripsi	Tipe & length	Null
Id	Variabel primary key	Int(2)	No
Nama_universitas	Variabel penampung nama universitas	Varchar(255)	No
Skor_qs	Variabel penampung skor QS General	Float	No
Peringkat_qs	Variabel penampung peringkat QS General	Int(2)	No

3.2.1.14 Tabel Peringkat QS Asia

Identifikasi : qs_asia

Deskripsi isi : tabel untuk menyimpan skor dan peringkat pemeringkatan QS rank versi region Asia

<u>Id_field</u>	Deskripsi	Tipe & length	Null
Id	Variabel primary key	Int(2)	No
Nama_universitas	Variabel penampung nama universitas	Varchar(255)	No
Skor_qsasia	Variabel penampung skor QS Asia	Float	No
Peringkat_qs-asia	Variabel penampung peringkat QS Asia	Int(2)	No

3.2.1.15 Tabel Peringkat QS Metode Helmholtz

Identifikasi : qs_helmholtz

Deskripsi isi : tabel untuk menyimpan skor dan peringkat pemeringkatan QS rank berdasarkan metode Helmholtz

Id_field	Deskripsi	Tipe & length	Null
Id	Variabel primary key	Int(2)	No
Nama_universitas	Variabel penampung nama universitas	Varchar(255)	No
Skor_helmholtz	Variabel penampung skor QS Helmholtz	Float	No
Peringkat_helmholtz	Variabel penampung peringkat QS Helmholtz	Int(2)	No

3.2.1.16 Tabel Peringkat QS Webometrics

Identifikasi : webometrics

Deskripsi isi : tabel untuk menyimpan skor dan peringkat pemeringkatan Webometrics

Id_field	Deskripsi	Tipe & length	Null
Id	Variabel primary key	Int(2)	No
Nama_universitas	Variabel penampung nama universitas	Varchar(255)	No
Skor_webometrics	Variabel penampung skor Webometrics	Float	No
Peringkat_webometrics	Variabel penampung peringkat Webometrics	Int(2)	No

3.2.2 Perancangan Antarmuka

Perangkat lunak yang dibangun berbasis pada aplikasi website. Antarmuka perangkat lunak dibangun menggunakan bahasa pemrograman PHP dan HTML. Berikut adalah rincian antarmuka yang dibangun dalam perangkat lunak.

3.2.2.1 Antarmuka Home

Identifikasi : antarmuka home (index)

Deskripsi isi : antarmuka ini berfungsi sebagai halaman root utama dalam perangkat lunak website. Terdapat satu kebutuhan fungsional dalam antarmuka ini yaitu pencarian perguruan tinggi berdasarkan potongan nama. Terdapat kolom search perguruan tinggi.

3.2.2.2 Antarmuka Latar Belakang Pemeringkatan

Identifikasi : antarmuka sekilas pemeringkatan

Deskripsi isi : antarmuka ini berfungsi sebagai penampil informasi sederhana yang menyatakan latar belakang kegunaan dari sistem pemeringkatan secara general.

3.2.2.3 Antarmuka Metode Penilaian

Identifikasi : antarmuka metode penilaian

Deskripsi isi : antarmuka ini berfungsi sebagai penampil informasi mengenai kriteria penilaian apa saja yang digunakan dalam sistem pemeringkatan. Terdapat informasi secara general dan informasi secara detail dari setiap tipe pemeringkatan, mulai dari pemeringkatan QS General, QS Asia, Webometrics, 4ICU, pemeringkatan DIKTI, dan QS menggunakan metode Helmholtz. Terdapat tombol untuk melihat tipe pemeringkatan yang diinginkan.

3.2.2.4 Antarmuka Peringkat Berdasarkan Region

Identifikasi : antarmuka peringkat berdasarkan region

Deskripsi isi : antarmuka ini berfungsi sebagai halaman pencarian peringkat perguruan tinggi berdasarkan region wilayah provinsi perguruan tinggi. Dengan menampilkan desain map negara Indonesia sebagai inputan pencarian yang mudah untuk dipahami. Pencarian dilakukan dengan cara menekan salah satu bagian provinsi dari map negara Indonesia.

3.2.2.5 Antarmuka Hasil Pencarian Berdasar Region Provinsi

Identifikasi : antarmuka hasil pencarian berdasarkan region provinsi

Deskripsi isi : antarmuka ini berfungsi sebagai halaman penampil hasil pencarian perguruan tinggi berdasarkan region provinsi yang dimiliki. Ditampilkan juga peringkat dari perguruan tinggi dalam hasil pencarian dari beberapa tipe pemeringkatan seperti QS General, QS Asia, Webometrics, 4ICU, pemeringkatan DIKTI, dan QS Helmholtz. Terdapat tombol untuk melihat tipe pemeringkatan yang diinginkan, dan terdapat kolom filter untuk mencari berdasarkan kota.

3.2.2.6 Antarmuka Hasil Pencarian Berdasar Pencarian Keyword Potongan Nama

Identifikasi : antarmuka hasil pencarian berdasarkan pencarian keyword potongan nama

Deskripsi isi : antarmuka ini berfungsi sebagai halaman penampil hasil pencarian perguruan tinggi berdasarkan keyword potongan nama. Terdapat informasi peringkat dari masing-masing perguruan tinggi yang dihasilkan. Terdapat tombol “Lihat Detail” untuk melihat detail informasi akademik perguruan tinggi.

3.2.2.7 Antarmuka Detail Informasi Akademik

Identifikasi : antarmuka detail informasi akademik

Deskripsi isi : antarmuka ini berfungsi sebagai halaman penampil informasi akademik perguruan tinggi. Terdapat beberapa informasi diantaranya mahasiswa, kecukupan dosen, status akreditasi, kualitas akademisi, produktifitas institusi, dan kepopuleran website. Ditampilkan juga grafik peringkat dari berbagai tipe pemeringkatan.

3.2.2.8 Antarmuka Pembobotan Pemeringkatan

Identifikasi : antarmuka pembobotan pemeringkatan

Deskripsi isi : antarmuka ini berfungsi sebagai halaman masukan bobot pemeringkatan yang diinginkan. Terdapat tabel tipe pemeringkatan dan mempunyai daftar kriteria yang digunakan. Pengguna dapat memasukkan bobot pemeringkatan sesuai kebutuhan. Nilai maksimal pembobotan dari satu tipe pemeringkatan adalah 100%, jika lebih dari itu maka akan muncul pesan error bobot berlebihan. Terdapat tombol pencarian berdasarkan tipe pemeringkatan, di antaranya “Cek QS-General”, “Cek QS-Asia”, “Cek 4ICU”, “Cek Webometrics”, “Cek DIKTI”, “Cek QS L-LDA”, “Cek QS PLSa”, “Cek QS Helmholtz”, “Cek QS Adaboost”.

3.2.2.9 Antarmuka Hasil Pemeringkatan Berdasar Pembobotan Tipe Pemeringkatan yang Diinginkan

Identifikasi : antarmuka hasil pemeringkatan berdasarkan hasil pembobotan

Deskripsi isi : antarmuka ini berfungsi sebagai halaman penampil hasil pembobotan perguruan tinggi sesuai dengan tipe pemeringkatan yang dikehendaki. Terdapat kolom filter untuk mencari berdasarkan kota.

3.2.3 Perancangan Antarmuka

Dalam menentukan peringkat perguruan tinggi menggunakan metode L-LDA, dibutuhkan beberapa kriteria

penilaian lain sebagai penunjang aspek kriteria kuantitatif dan kualitatif [4]. Penjelasan masing-masing kriteria akan dijelaskan dibawah ini.

3.2.3.1 Reputasi Akademik

Merupakan kriteria utama yang bersifat kualitatif yang didapatkan dari proses teks mining dokumen abstrak perguruan tinggi menggunakan metode Helmholtz dan metode Labeled Latent Dirichlet Allocation (L-LDA) [6]. Mempunyai bobot pemeringkatan 40% dari total bobot keseluruhan. Alur proses menghasilkan kriteria reputasi akademik seperti pada Gambar 3.9.



Gambar 3.9 Alur Menentukan Reputasi Akademik

Langkah pertama dengan mengumpulkan dataset berupa abstrak paper perguruan tinggi, kemudian membangun corpus keyword yang didapatkan dari taxonomy Bloom dan dikembangkan menjadi corpus TKT seperti pada langkah 4 pada bagan diatas. Selanjutnya melakukan tahap teks preprocessing untuk menghasilkan kata inputan yang baku dari dokumen dataset. Hasil dari tahap teks preprocessing akan digunakan sebagai penentuan asumsi label topik awal, dimana setiap dokumen harus mempunyai label sebagai acuan topiknya. Kemudian masuk pada tahap inferensi Gibbs Sampling untuk menghasilkan probabilitas terbaik dari dokumen terhadap topik. Hasil dari tahap ini berupa probabilitas akhir yang nantinya akan dihitung menggunakan pembobotan final reputasi akademik. Hasil ini kemudian disimpan dalam database sebagai nilai indicator reputasi akademik.

3.2.3.2 Akreditasi Perguruan Tinggi

Merupakan kriteria yang bersifat kuantitatif. Didapatkan dari proses grabbing data online pada website BAN-PT. Data yang diambil adalah data nama perguruan tinggi, masa berlaku, nilai huruf akreditasi, serta status perguruan tinggi. Mempunyai bobot 10% dari total bobot keseluruhan. Alur proses menghasilkan kriteria akreditasi perguruan tinggi seperti pada Gambar 3.10.



Gambar 3.10 Alur Menentukan Skor Akreditasi

Langkah pertama dengan menyiapkan situs website yang akan dijadikan sasaran *grabbing*, dalam Tugas Akhir ini menggunakan 2 percobaan pada website BAN-PT dan website penyedia informasi akreditasi. Kemudian Menentukan teknik grabbing apa dan library yang akan digunakan, dalam Tugas Akhir ini menggunakan Bahasa pemrograman PHP, library HTML DOM Parser, dan beberapa fungsi bawahan dari PHP sendiri. Kemudian melakukan grabbing dengan ketentuan harus mempunyai bandwidth yang cukup. Karena proses grabbing membutuhkan waktu yang cukup lama melihat konten yang ingin diambil cukup besar dan banyak. Dalam proses grabbing data yang masuk akan langsung disimpan dalam table temporary pada database, sebagai penampung sementara karena tujuan utama disini adalah mengambil sebisa mungkin. Dalam database table temporary ini kemudian akan dianalisa dan dilakukan normalisasi hasil untuk menghasilkan skor akreditasi yang paling akhir. Skor akreditasi ini kemudian akan disimpan lagi dalam database dengan table yang berbeda sebagai implementasi skor final akreditasi perguruan tinggi.

3.2.3.3 Jumlah Rasio Mahasiswa dan Tenaga Pengajar Aktif

Merupakan kriteria yang bersifat kuantitatif. Didapatkan dari data rekap database online PDDIKTI. Data yang diambil merupakan rekap yang berisi jumlah mahasiswa, nama perguruan tinggi serta jumlah tenaga pengajar keseluruhan. Mempunyai bobot 20% dari total bobot keseluruhan. Alur proses menghasilkan kriteria rasio mahasiswa dan tenaga pengajar aktif seperti pada Gambar 3.2-3.

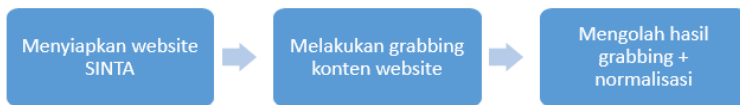


Alur Menentukan Skor Rasio Mahasiswa dan Tenaga Penagajar Aktif

Langkah pertama dengan mengumpulkan data rakap jumlah mahasiswa dan rekap jumlah tenaga pengajar dari seluruh perguruan tinggi di Indonesia. Data ini disediakan oleh PDDIKTI sebagai pusat databasenya. Untuk mempermudah proses analisa, dalam Tugas Akhir ini menggunakan data rekap berformat .xlsx kemudian akan dibaca menggunakan library excel dalam Bahasa pemrograman python. Karena file excel berupa row dan column maka akan diambil beberapa variable penting sebagai acuan penilaian. Variable jumlah mahasiswa semester 1 tahun 1, mahasiswa semester 2 tahun 1, mahasiswa semester 1 tahun 2, dan semester 2 tahun 2 menjadi acuan penilaian mahasiswa. Serta data jumlah teanga pengajar secara keseluruhan dari setiap perguruan tinggi akan menjdi acuan penilaian jumlah tenaga pangajar. Nilai dari variable ini akan dinormalisasi terlebih dahulu dengan membagi dengan nilai tertinggi. Kemudian akan dirasiokan dengan jumlah tenaga pengajar, kemudian disimpan dalam database sebagai implementasi skor final rasio mahasiswa dan tenaga pengajar aktif.

3.2.3.4 Jumlah Sitasi Akademisi Penelitian

Merupakan kriteria yang bersifat kuantitatif. Didapatkan dari proses grabbing data online Science and Technology Index (SINTA). Data yang digunakan adalah nama perguruan tinggi, serta jumlah sitasi dan author akademisi yang terindex SINTA. Mempunyai bobot 20% dari total bobot keseluruhan. Alur proses menghasilkan kriteria jumlah sitasi akademisi penelitian seperti pada Gambar 3.2-4.



Alur Menentukan Skor Jumlah Sitasi Akademisi Penelitian

Langkah pertama dengan menyiapkan website sasaran grabbing yaitu situs website Science and Technology Index (SINTA) dari Kemenristek Dikti. Dalam situs ini terdapat beberapa variable penting sebagai acuan penilaian indikator. Setiap variable akan dilakukan teknik *grabbing* dan sama seperti pada proses menentukan indikator akreditasi perguruan tinggi. Dalam proses ini juga membutuhkan database table temporary sebagai penampung sementara. Variable yang digunakan adalah jumlah sitasi dari google scholar dan scopus. Kemudian dari table temporary ini akan dianalisa dan dinormalisasi dan dicari rata-ratanya sebagai skor final jumlah sitasi akademisi penelitian dan disimpan lagi dalam table final.

3.2.3.5 Jumlah Tenaga Pengajar (Dosen)

Merupakan kriteria yang bersifat kuantitatif yang didapatkan dari data rekap database online PDDIKTI. Data yang diambil merupakan jumlah tenaga pengajar dengan berbagai status ketenaga kerjaan, seperti pegawai tetap maupun tidak tetap. Mempunyai bobot 5% dari total bobot keseluruhan. Alur proses

menghasilkan kriteria jumlah tenaga pengajar (dosen) seperti pada Gambar 3.2-3 diatas.

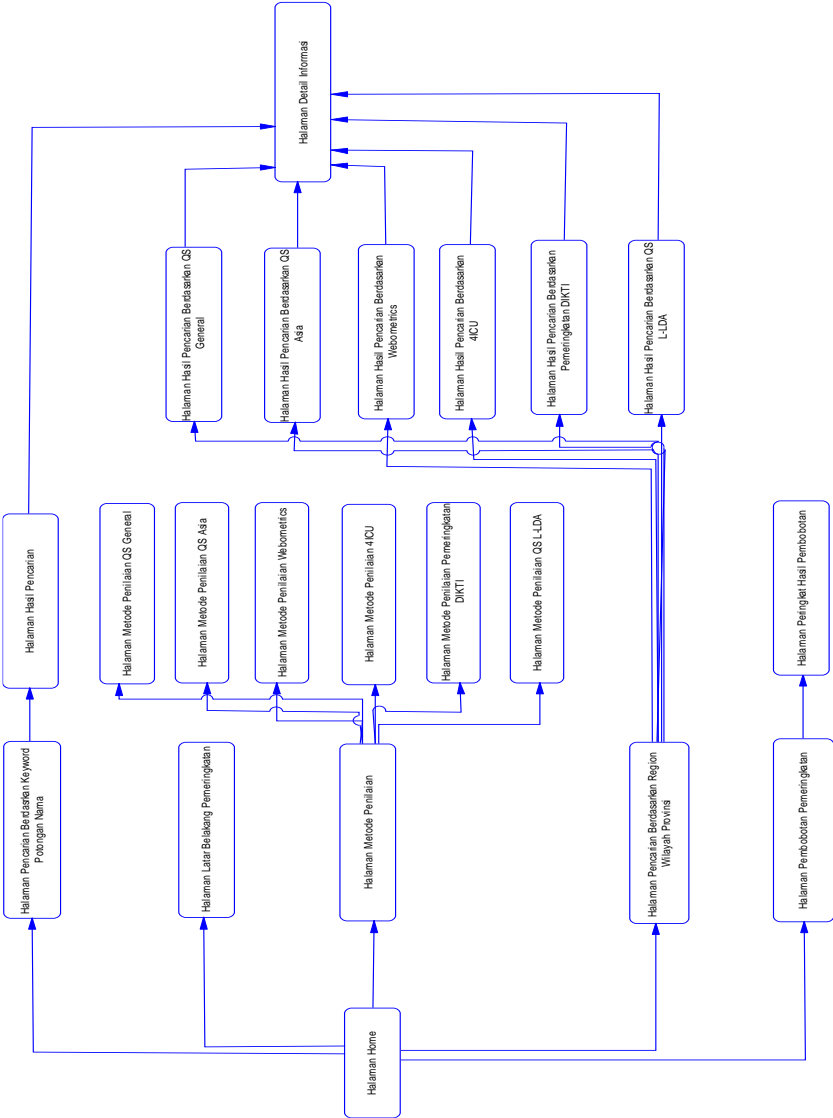
Prinsip kerja Menentukan indikator ini sama seperti indikator jumlah rasio mahasiswa dan tenaga pengajar aktif. Hanya saja disini ada beberapa data rekap yang digunakan sebagai pendukung yaitu rekap jumlah tenaga pengajar yang bersifat tetap dan tidak tetap (honorer). Data ini akan dibaca menggunakan library excel pada Bahasa pemrograman PHP kemudian akan dianalisa dan dinormalisasi untuk menghasilkan jangkauan nilai yang rata. Terdapat variable tanga pengajar tetap dan tidak tetap yang nilai masing-masing telah dinormalisasi, kemudian dijumlahkan dan dirata-rata. Nilai rata-rata ini akan menjadi skor indikator jumlah tenaga pengajar (dosen).

3.2.3.6 Kualitas Sitasi

Merupakan kriteria yang bersifat kuantitatif yang didapatkan dari proses grabbing data online Science and Technology Index (SINTA). Data yang diambil merupakan nilai h-index dan i10-index Google Scholar dari akademisi perguruan tinggi. Mempunyai bobot 5% dari total bobot keseluruhan. Alur proses menghasilkan kriteria kualitas sitasi seperti pada Gambar 3.2-4 diatas.

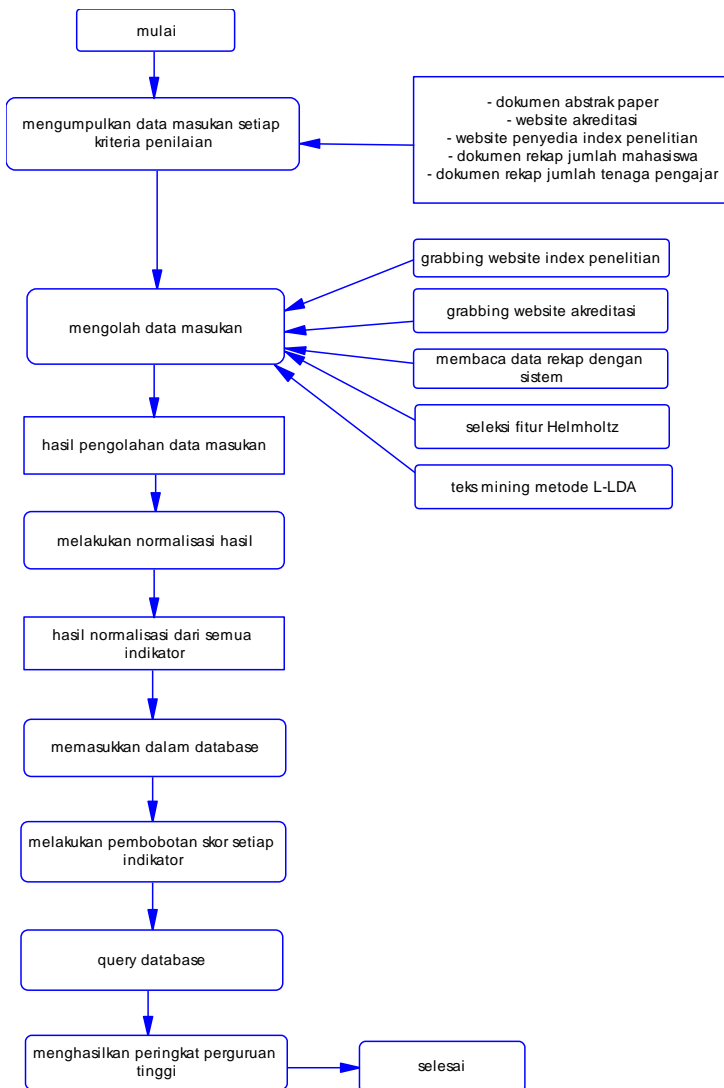
Prinsip kerja dalam Menentukan indikator ini sama seperti pada indikator jumlah sitasi akademisi penelitian. Hanya saja disini yang diambil adalah variable jumlah h-index dan i10-index dari google scholar. Kedua variable ini akan dilakukan normalisasi nilai dan dicari rata-ratanya. Dan kemudian disimpan dalam table database baru sebagai implementasi skor indikator kualitas sitasi.

3.2.4 **Proses Sistem Aplikasi Pengguna (Diagram Aplikasi)**



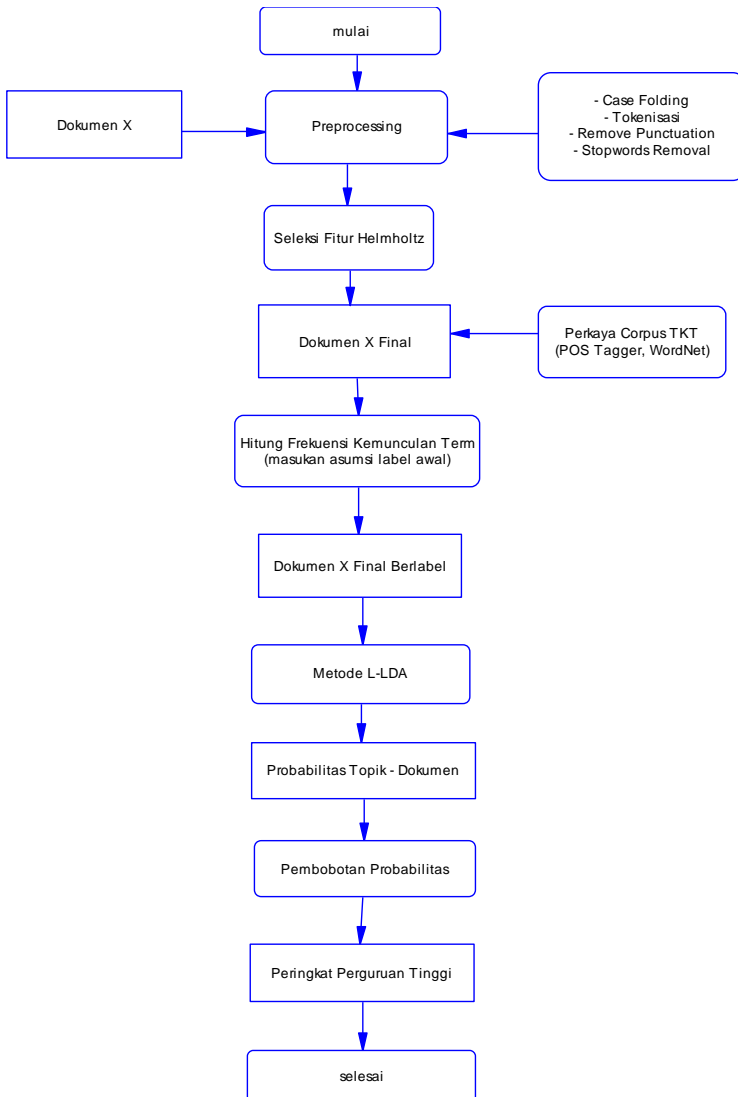
Gambar 3.11 Diagram Aplikasi

3.2.5 Proses Kerja Sistem Secara Keseluruhan (Flowchart Sistem)



Gambar 3.12 Flowchart Sistem Keseluruhan

3.2.6 Proses Teks Mining Metode Helmholtz (Flowchart Metode Helmholtz)



Gambar 3.13 Flowchart Metode Helmholtz

BAB IV IMPLEMENTASI SISTEM

Bab ini membahas implementasi dari perancangan sistem pemeringkatan universitas. Setiap dijelaskan dalam bentuk kode. Implementasi perangkat lunak dibedakan menjadi beberapa subbab, yaitu lingkungan implementasi, tahap prapemrosesan, tahap perhitungan probabilitas kata, topik dan dokumen, tahap pengelompokkan kata dan dokumen, tahap pelabelan topik, tahap pembobotan dan *screenshot* hasil implementasi antarmuka.

4.1 Lingkungan Implementasi

Lingkungan pengembangan sistem yang digunakan untuk mengembangkan Tugas Akhir ini dilakukan pada lingkungan dan kaskas sebagai berikut.

1. *Database* yang digunakan pada server adalah MySQL.
2. 1 PC untuk pengembangan *User* menggunakan Xeon E5-2620 v1212 Cores x 2.1 GHZ
3. 1 PC untuk uji coba menggunakan Xeon E5-2620 v1212 Cores x 2.1 GHZ
4. StarUML untuk pembuatan diagram, Sublime sebagai teks editor.
5. Google Chrome 58.0.3 sebagai antarmuka untuk pengujian aplikasi klien.

4.2 Tahap Implementasi

Pada bagian ini terdapat beberapa subproses di dalamnya. Prapemrosesan merupakan tahap dataset yang paling awal. Kemudian, tahap selanjutnya yaitu Bagian ini terdiri dari pengambilan kata-kata, tokenisasi, dan *stopwords* serta *stemming*.

4.2.1 Tahap Prapemrosesan

Pada tahap ini, program akan melakukan prapemrosesan dokumen yang terdiri dari pengambilan kata-kata, tokenisasi, dan *stopwords* serta *stemming*. Hal ini dilakukan untuk membantu proses *text mining* selanjutnya.

4.2.2 **Memperkaya Corpus**

Pada tahap ini, program akan melakukan proses untuk memperkaya corpus TKT. POS-Tagging menjadi salah satu metode yang digunakan. POS-Tagging akan mengambil kata dengan indeks $i-1$ dan $i+1$. Corpus ini digunakan sebagai patokan dalam proses melakukan pencarian persamaan atau sinonim kata dari dokumen corpus TKT satu terhadap dokumen corpus TKT lainnya. Persamaan tersebut akan menghasilkan kata-kata dengan makna yang sama untuk dibagikan dan dimasukkan ke dalam TKT yang bersangkutan.

Selain itu, digunakan pula WordNet. Pertama-tama, dilakukan pencarian sinonim dari corpus TKT terhadap basis data bahasa WordNet. Sinonim kata yang terdapat pada WordNet akan ditambahkan ke dalam corpus TKT. Setiap sinonim kata memiliki penanda jika sinonim tersebut dekat atau jauh. 0 menandakan dekat dan seterusnya. Pada Tugas Akhir ini, saya melakukan percobaan dengan menggunakan WordNet level 1, level 2, level 3, dan level all.

4.3 **Tahap Fitur Seleksi TKT**

Pada bagian ini, akan dibahas mengenai implementasi fitur seleksi menggunakan *Helmholtz* terhadap TKT. Perhitungan ini dilakukan agar masing-masing kata dapat diketahui nilai kepentingannya masing-masing. Kemudian, nilai yang tidak bernilai sesuai dengan aturan akan dibuang. Metode ini berguna dalam mengurangi kata atau fitur yang dianggap tidak penting. Masing-masing kata akan memiliki nilai-nilainya masing-masing untuk mengukur mana kata yang penting dan tidak seperti pada Gambar 4.1.


```

Dokumen 1
Word: kita, NFA: 0.06301
Word: lelah, NFA: 0.04201
Word: ganggu, NFA: 0.04201
Dokumen 2
Word: motivasi, NFA: 0.13863
Word: ajar, NFA: 0.11552
Word: semangat, NFA: 0.04621
Dokumen 3
Word: partai, NFA: 0.10108
Word: politik, NFA: 0.10108
Word: tingkat, NFA: 0.04332
Dokumen 4
Word: kerja, NFA: 0.10548
Word: anggur, NFA: 0.06027
Word: masalah, NFA: 0.03014

```

Gambar 4.1 Output Helmholtz

4.4 Klasifikasi *Term Frequency*

Sebelum masuk ke dalam metode L-LDA, akan dihitung terlebih dahulu *term frequency* dari abstrak. Pada tahap ini, akan dipilih tiga topik TKT tertinggi untuk dilanjutkan ke dalam metode L-LDA. Tiga topik tersebut berdasarkan atas banyaknya kata yang masuk ke dalam TKT masing-masing level).

4.5 Penentuan Nilai Akhir Peringkat Universitas

Terakhir, akan dilakukan penentuan nilai akhir universitas dengan menggunakan bobot yang telah ditentukan. Bobot perhitungan final khusus untuk reputasi akademik yakni dengan menggunakan rumus (11).

$$Bobot\ final = \frac{\sum (\vartheta^{(d_i)} \times bobot\ level(i))}{\sum topik\ keseluruhan} \quad (11)$$

Selanjutnya, akan dipilih tiga label teratas sebagai variabel penting dalam menentukan nilai akhir. Masing-masing label akan dikalikan dengan bobot level dari masing-masing level TKT. Bobot level yang digunakan pada rumus (4) beragam menyesuaikan dengan masing-masing level TKT. Berikut adalah tabel rincian masing-masing bobot setiap TKT seperti pada Tabel 4.1.

Tabel 4.1 Bobot Level TKT

No.	Topik Final	Bobot Level
1.	TKT level 1	10
2.	TKT level 2	20
3.	TKT level 3	30
4.	TKT level 4	40
5.	TKT level 5	50
6.	TKT level 6	60
7.	TKT level 7	70
8.	TKT level 8	80
9.	TKT level 9	90

Setelah hasil berhasil dihitung, nilai reputasi akademik dikalikan dengan persentase berdasarkan dengan aturan QS. Selain reputasi akademik, terdapat juga reputasi karyawan, jumlah mahasiswa aktif, jumlah sitasi akademisi, jumlah tenaga pengajar, serta kualitas sitasi seperti indeks-H dan indeks-i10. Setelah dikalikan oleh sistem, peringkat masing-masing universitas akan muncul beserta nilai akhirnya.

4.6 Implementasi Basis Data

4.6.1 Create Database “database_qs_rank”

```
CREATE DATABASE /*!32312 IF NOT
EXISTS*/`database_qs_2` /*!40100 DEFAULT CHARACTER SET
latin1 */;

USE `database_qs_2`;
```

Kode Sumber 5 Kode Sumber Create Database

4.6.2 Create Table “universitas”

```
CREATE TABLE `universitas` (
  `id` int(2) unsigned NOT NULL AUTO_INCREMENT,
  `nama` varchar(255) NOT NULL,
  `domain` varchar(150) NOT NULL,
  `url_web` varchar(150) NOT NULL,
  `kota` varchar(150) NOT NULL,
  `provinsi` varchar(150) NOT NULL,
  PRIMARY KEY (`id`)
) ENGINE=InnoDB AUTO_INCREMENT=410 DEFAULT
CHARSET=latin1;
```

Kode Sumber 6 Kode Sumber Create Table universitas

4.6.3 Create Table “reputasi akademik”

```
CREATE TABLE `reputasi_akademik` (
  `id` int(2) unsigned NOT NULL AUTO_INCREMENT,
  `nama_universitas` varchar(255) NOT NULL,
  `skor_universitas` float NOT NULL,
  `bobot_total` int(2) NOT NULL,
  `jumlah_dokumen` int(2) NOT NULL,
  `tanggal_akses` datetime NOT NULL,
  PRIMARY KEY (`id`)
) ENGINE=InnoDB AUTO_INCREMENT=484 DEFAULT
CHARSET=latin1;
```

Kode Sumber 7 Kode Sumber Create Table reputasi_akademik

4.6.4 Create Tabel “akreditasi”

```
CREATE TABLE `tampung_akreditasi` (
  `id` int(2) unsigned NOT NULL AUTO_INCREMENT,
  `nomor` int(3) NOT NULL,
  `nama_universitas` varchar(255) NOT NULL,
  `peringkat` varchar(150) NOT NULL,
  `masa_berlaku` varchar(150) NOT NULL,
  `status` varchar(150) NOT NULL,
  `skor` int(2) NOT NULL,
  `tmp_nama_universitas` text NOT NULL,
  PRIMARY KEY (`id`)
) ENGINE=InnoDB AUTO_INCREMENT=13095
DEFAULT CHARSET=latin1;
```

Kode Sumber 8 Kode Sumber Create Table akreditasi

4.6.5 Create Tabel “jumlah_tenaga_pengajar_jabatan_akademik”

```
CREATE TABLE
`jumlah_tenaga_pengajar_jabatan_akademik` (
  `id` int(2) NOT NULL AUTO_INCREMENT,
  `kode_universitas` int(2) NOT NULL,
  `nama_universitas` varchar(100) NOT NULL,
  `asisten_ahli_10000` int(2) NOT NULL,
  `asisten_ahli_15000` int(2) NOT NULL,
  `lektor_20000` int(2) NOT NULL,
  `lektor_30000` int(2) NOT NULL,
  `lektor_kepala_40000` int(2) NOT NULL,
  `lektor_kepala_55000` int(2) NOT NULL,
  `lektor_kepala_70000` int(2) NOT NULL,
  `profesor_85000` int(2) NOT NULL,
  `profesor_105000` int(2) NOT NULL,
  `tanpa_jabatan` int(2) NOT NULL,
  `status_universitas` tinytext NOT NULL,
  PRIMARY KEY (`id`)
) ENGINE=InnoDB AUTO_INCREMENT=16840 DEFAULT
CHARSET=latin1;
```

Kode Sumber 9 Kode Sumber Create Table
jumlah_tenaga_pengajar_jabatan_akademik

4.6.6 Create Tabel “jumlah_mahasiswa”

```
CREATE TABLE `jumlah_mahasiswa` (
  `id` int(2) unsigned NOT NULL AUTO_INCREMENT,
  `kode_universitas` int(4) NOT NULL,
  `nama_universitas` varchar(255) NOT NULL,
  `jumlah_tahun1_semester1` int(2) NOT NULL,
```

Kode Sumber 10 Kode Sumber Create Table jumlah_mahasiswa

4.6.7 Create Tabel “jumlah_tenaga_pengajar_tetap”

```
CREATE TABLE `jumlah_tenaga_pengajar_tetap` (
  `id` int(2) NOT NULL AUTO_INCREMENT,
  `kode_universitas` int(2) NOT NULL,
  `nama_universitas` varchar(100) NOT NULL,
  `D1` int(2) NOT NULL,
  `D2` int(2) NOT NULL,
  `D3` int(2) NOT NULL,
  `D4` int(2) NOT NULL,
  `S1` int(2) NOT NULL,
  `S2` int(2) NOT NULL,
  `S3` int(2) NOT NULL,
  `non_formal` int(2) NOT NULL,
  `informal` int(2) NOT NULL,
  `lainnya` int(2) DEFAULT NULL,
  `SP1` int(2) NOT NULL,
  `SP2` int(2) NOT NULL,
  `profesi` int(2) NOT NULL,
  `tanpa_jenjang` int(2) NOT NULL,
  `status_universitas` tinytext NOT NULL,
  PRIMARY KEY (`id`)
) ENGINE=InnoDB AUTO_INCREMENT=59738 DEFAULT
CHARSET=latin1;
```

Kode Sumber 11 Kode Sumber Create Table
jumlah_tenaga_pengajar_tetap

4.6.8 Create Tabel “jumlah_tenaga_pengajar_tidak_tetap”

```
CREATE TABLE `jumlah_tenaga_pengajar_tidak_tetap`
(
  `id` int(2) NOT NULL AUTO_INCREMENT,
  `kode_universitas` int(2) NOT NULL,
  `nama_universitas` varchar(100) NOT NULL,
  `D1` int(2) NOT NULL,
  `D2` int(2) NOT NULL,
  `D3` int(2) NOT NULL,
  `D4` int(2) NOT NULL,
  `S1` int(2) NOT NULL,
  `S2` int(2) NOT NULL,
  `S3` int(2) NOT NULL,
  `non_formal` int(2) NOT NULL,
  `informal` int(2) NOT NULL,
  `lainnya` int(2) DEFAULT NULL,
  `SP1` int(2) NOT NULL,
  `SP2` int(2) NOT NULL,
  `profesi` int(2) NOT NULL,
  `tanpa_jenjang` int(2) NOT NULL,
  `status_universitas` tinytext NOT NULL,
  PRIMARY KEY (`id`)
) ENGINE=InnoDB AUTO_INCREMENT=8885 DEFAULT
CHARSET=latin1;
```

Kode Sumber 12 Kode Sumber Create Table
jumlah_tenaga_pengajar_tidak_tetap

4.6.9 Create Tabel “jumlah_sitasi_sinta”

```
CREATE TABLE `jumlah_sitasi_sinta` (
  `id` int(2) unsigned NOT NULL AUTO_INCREMENT,
  `nama_universitas` varchar(255) NOT NULL,
  `jumlah_author_scholar` int(2) NOT NULL,
  `jumlah_paper_scholar` int(2) NOT NULL,
  `jumlah_sitasi_scholar` int(2) NOT NULL,
  `jumlah_jurnal_artikel` int(2) NOT NULL,
  `jumlah_book_chapter` int(2) NOT NULL,
  `jumlah_paper_conference` int(2) NOT NULL,
  `jumlah_artikel_scopus` int(2) NOT NULL,
  `jumlah_non_artikel_scopus` int(2) NOT NULL,
  `jumlah_sitasi_scopus` int(2) NOT NULL,
  `sinta_score` int(2) NOT NULL,
  PRIMARY KEY (`id`)
) ENGINE=InnoDB AUTO_INCREMENT=4006 DEFAULT
CHARSET=latin1;
```

Kode Sumber 13 Kode Sumber Create Table jumlah_sitasi_sinta

4.6.10 Create Tabel “jumlah_author_sinta”

```
CREATE TABLE `jumlah_author_sinta` (
  `id` int(2) unsigned NOT NULL AUTO_INCREMENT,
  `id_data` int(2) NOT NULL,
  `nomor` int(2) NOT NULL,
  `nama_author` varchar(150) NOT NULL,
  `nama_universitas` varchar(150) NOT NULL,
  `jumlah_sitasi_scholar` int(2) NOT NULL,
  `i10_index_scholar` int(2) NOT NULL,
  `h_index_scholar` int(2) NOT NULL,
  `waktu_akses` tinytext NOT NULL,
  PRIMARY KEY (`id`)
) ENGINE=InnoDB AUTO_INCREMENT=2189 DEFAULT
CHARSET=latin1;
```

Kode Sumber 14 Kode Sumber Create Table jumlah_author_sinta

4.6.11 Create Tabel “presence”

```
CREATE TABLE `presence` (
  `id` int(2) unsigned NOT NULL AUTO_INCREMENT,
  `url_web` varchar(255) NOT NULL,
  `domain_search` varchar(150) NOT NULL,
  `hasil_search` int(2) NOT NULL,
  `tanggal_akses` datetime NOT NULL,
  PRIMARY KEY (`id`)
) ENGINE=InnoDB AUTO_INCREMENT=526 DEFAULT
CHARSET=latin1;
```

Kode Sumber 15 Kode Sumber Create Table presence

4.6.12 Create Tabel “impact”

```
CREATE TABLE `impact` (
  `id` int(2) unsigned NOT NULL AUTO_INCREMENT,
  `url_web` varchar(255) NOT NULL,
  `domain_web` varchar(150) NOT NULL,
  `page_authority` int(2) NOT NULL,
  `domain_authority` int(2) NOT NULL,
  `backlink_eksternal` int(2) NOT NULL,
  `tanggal_akses` datetime NOT NULL,
  PRIMARY KEY (`id`)
) ENGINE=InnoDB AUTO_INCREMENT=1960 DEFAULT
CHARSET=latin1;
```

Kode Sumber 16 Kode Sumber Create Table impact

4.6.13 Create Tabel “alexa_rank”

```
CREATE TABLE `alexa_rank` (
  `id` int(2) unsigned NOT NULL AUTO_INCREMENT,
  `url_web` varchar(255) NOT NULL,
  `rank_world` int(2) NOT NULL,
  `rank_country` int(2) NOT NULL,
  `country_name` varchar(150) NOT NULL,
  `tanggal_akses` datetime NOT NULL,
  PRIMARY KEY (`id`)
) ENGINE=InnoDB AUTO_INCREMENT=444 DEFAULT
CHARSET=latin1;
```

Kode Sumber 17 Kode Sumber Create Table alexa_rank

4.6.14 Create Tabel “kumpulan_skor_final”

```
CREATE TABLE `kumpulan_skor_final` (
  `id` int(2) unsigned NOT NULL AUTO_INCREMENT,
  `kode_universitas` int(3) NOT NULL,
  `nama_universitas` varchar(255) NOT NULL,
  `skor_reputasi_akademik_normalisasi` float NOT
NULL,
  `skor_nilai_akreditasi_normalisasi` float NOT
NULL,
  `skor_jumlah_mahasiswa_normalisasi` float NOT
NULL,
  `skor_jumlah_sitasi_normalisasi` float NOT NULL,
  `skor_jumlah_tenaga_pengajar_normalisasi` float
NOT NULL,
  `skor_kualitas_sitasi_normalisasi` float NOT NULL,
  `skor_jumlah_phd_normalisasi` float NOT NULL,
  `skor_jumlah_paper_normalisasi` float NOT NULL,
  `skor_jumlah_jurnal_artikel` float NOT NULL,
  `skor_jumlah_paper_conference` float NOT NULL,
  `skor_precense_normalisasi` float NOT NULL,
  `skor_impact_normalisasi` float NOT NULL,
  `skor_transparency_normalisasi` float NOT NULL,
  `skor_excellence_normalisasi` float NOT NULL,
  `skor_moz_analytic_normalisasi` float NOT NULL,
  `skor_alex_rank_normalisasi` float NOT NULL,
  `skor_kualitas_dosen_normalisasi` float NOT NULL,
  `skor_kecukupan_dosen_normalisasi` float NOT NULL,
  `skor_kualitas_manajemen_normalisasi` float NOT
NULL,
  `skor_kualitas_kegiatan_kemahasiswaan_normalisasi`
float NOT NULL,
  `skor_kualitas_kegiatan_penelitian_normalisasi`
float NOT NULL,
  PRIMARY KEY (`id`)
) ENGINE=InnoDB AUTO_INCREMENT=11356
DEFAULT CHARSET=latin1;
```

Kode Sumber 18 Kode Sumber Create Table
kumpulan_skor_final

4.6.15 Create Tabel “4icu”

```
CREATE TABLE `4icu` (
  `id` int(2) unsigned NOT NULL AUTO_INCREMENT,
  `nama_universitas` varchar(255) NOT NULL,
  `skor_4icu` float NOT NULL,
  `peringkat_4icu` int(2) NOT NULL,
  PRIMARY KEY (`id`)
) ENGINE=InnoDB AUTO_INCREMENT=2128 DEFAULT
CHARSET=latin1;
```

Kode Sumber 19 Kode Sumber Create Table 4icu

4.6.16 Create Tabel “dikti”

```
CREATE TABLE `dikti` (
  `id` int(2) unsigned NOT NULL AUTO_INCREMENT,
  `nama_universitas` varchar(255) NOT NULL,
  `skor_dikti` float NOT NULL,
  `peringkat_dikti` int(2) NOT NULL,
  PRIMARY KEY (`id`)
) ENGINE=InnoDB AUTO_INCREMENT=1773 DEFAULT
CHARSET=latin1;
```

Kode Sumber 20 Kode Sumber Create Table dikti

4.6.17 Create Tabel “qs_general”

```
CREATE TABLE `qs_general` (
  `id` int(2) unsigned NOT NULL AUTO_INCREMENT,
  `nama_universitas` varchar(255) NOT NULL,
  `skor_qs` float NOT NULL,
  `peringkat_qs` int(2) NOT NULL,
  PRIMARY KEY (`id`)
) ENGINE=InnoDB AUTO_INCREMENT=2838 DEFAULT
CHARSET=latin1;
```

Kode Sumber 21 Kode Sumber Create Table qs_general

4.6.18 Create Tabel “qs_asia”

```
CREATE TABLE `qs_asia` (
  `id` int(2) unsigned NOT NULL AUTO_INCREMENT,
  `nama_universitas` varchar(255) NOT NULL,
  `skor_qs_asia` float NOT NULL,
  `peringkat_qs_asia` int(2) NOT NULL,
  PRIMARY KEY (`id`)
) ENGINE=InnoDB AUTO_INCREMENT=2838 DEFAULT
CHARSET=latin1;
```

Kode Sumber 22 Kode Sumber Create Table qs_asia

4.6.19 Create Tabel “qs_helmholtz”

```
CREATE TABLE `qs_llda` (
  `id` int(2) unsigned NOT NULL AUTO_INCREMENT,
  `nama_universitas` varchar(255) NOT NULL,
  `skor_qs_llda` float NOT NULL,
  `peringkat_qs_helmholtz` int(2) NOT NULL,
  PRIMARY KEY (`id`)
) ENGINE=InnoDB AUTO_INCREMENT=3193 DEFAULT
CHARSET=latin1;
```

Kode Sumber 23 Kode Sumber Create Table qs_helmholtz

4.6.20 Create Tabel “webometrics”

```
CREATE TABLE `webometrics` (
  `id` int(2) unsigned NOT NULL AUTO_INCREMENT,
  `nama_universitas` varchar(255) NOT NULL,
  `skor_webometrics` float NOT NULL,
  `peringkat_webometrics` int(2) NOT NULL,
  PRIMARY KEY (`id`)
) ENGINE=InnoDB AUTO_INCREMENT=2838 DEFAULT
CHARSET=latin1;
```

Kode Sumber 24 Kode Sumber Create Table webometrics

4.7 Antarmuka Halaman Utama

Berikut merupakan antarmuka halaman utama situs pemeringkatan universitas. Halaman utama memperlihatkan kebutuhan fungsional yang ditawarkan oleh situs tersebut.

4.7.1 Antarmuka Kolom *Search*

Berikut merupakan tampilan antarmuka kolom *search* pada halaman utama situs. Kolom *search* berfungsi untuk mencari universitas berdasarkan namanya seperti terlihat pada Gambar 4.2.



Gambar 4.2 Antarmuka Kolom Search

4.7.2 Antarmuka Peringkat Berdasarkan *Region*

Berikut ini merupakan tampilan antarmuka peringkat berdasarkan *region* pada halaman utama situs. Ini berguna untuk mencari peringkat berdasarkan wilayah provinsi atau *region* seperti terlihat pada Gambar 4.3.



Gambar 4.3 Antarmuka Peringkat Berdasarkan Region

4.7.3 **Antarmuka Atur Bobot Pemeringkatan**

Berikut ini merupakan tampilan antarmuka atur bobot pemeringkatan pada halaman utama situs. Ini berguna untuk mengatur bobot. Terdapat dua jenis pengaturan, yakni *default* dan *custom*. *Default* merupakan bobot dasar yang tidak dapat diganti-ganti. Sedangkan, *custom* merupakan bobot yang dapat diganti-ganti oleh pengunjung seperti terlihat pada Gambar 4.4.

Atur Bobot Pemeringkatan						
Default Custom						
Indikator	QS(General)	QS Asia	Webometrics	4ICU	Sinta	Pemeringkatan Dikti
Reputasi Akademik	<input type="text" value="40"/>	<input type="text" value="30"/>	<input type="text" value="0"/>	<input type="text" value="0"/>	<input type="text" value="0"/>	<input type="text" value="0"/>
Reputasi Employer	<input type="text" value="10"/>	<input type="text" value="20"/>	<input type="text" value="0"/>	<input type="text" value="0"/>	<input type="text" value="0"/>	<input type="text" value="0"/>
Jumlah Mahasiswa Aktif	<input type="text" value="20"/>	<input type="text" value="15"/>	<input type="text" value="0"/>	<input type="text" value="0"/>	<input type="text" value="0"/>	<input type="text" value="0"/>
Jumlah Sitasi Akademisi	<input type="text" value="20"/>	<input type="text" value="10"/>	<input type="text" value="0"/>	<input type="text" value="0"/>	<input type="text" value="0"/>	<input type="text" value="0"/>
Jumlah Tenaga Pengajar	<input type="text" value="5"/>	<input type="text" value="2.5"/>	<input type="text" value="0"/>	<input type="text" value="0"/>	<input type="text" value="0"/>	<input type="text" value="0"/>
Kualitas Sitasi (H-index & I10-index)	<input type="text" value="5"/>	<input type="text" value="2.5"/>	<input type="text" value="0"/>	<input type="text" value="0"/>	<input type="text" value="0"/>	<input type="text" value="0"/>
Jumlah PhD (S3)	<input type="text" value="0"/>	<input type="text" value="5"/>	<input type="text" value="0"/>	<input type="text" value="0"/>	<input type="text" value="0"/>	<input type="text" value="0"/>

Gambar 4.4 Antarmuka Atur Bobot Pemeringkatan

4.7.4 **Antarmuka Daftar Peringkat Universitas**

Berikut ini merupakan tampilan antarmuka daftar peringkat universitas secara terurut setelah dikalkulasi menggunakan bobot yang dimasukkan seperti terlihat pada Gambar 4.5.

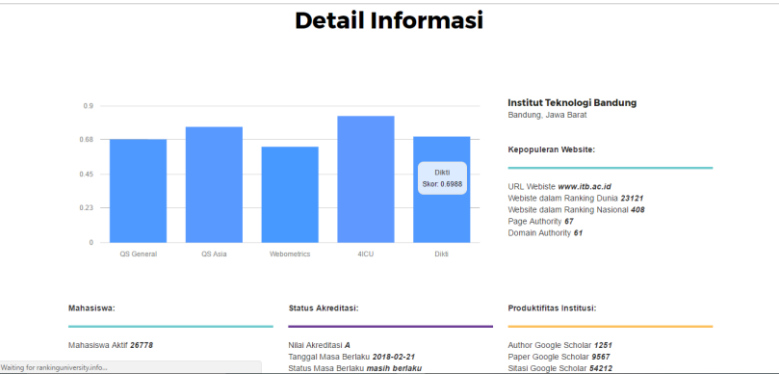
Peringkat Perguruan Tinggi (QS)

No.	Nama Perguruan Tinggi	Skor	Kota	Provinsi
1	Institut Teknologi Bandung	0.6801	Bandung	Jawa Barat
2	Universitas Gadjah Mada	0.6522	Yogyakarta	DI Yogyakarta
3	Universitas Indonesia	0.6414	Depok	Jawa Barat
4	Institut Pertanian Bogor	0.5673	Bogor	Jawa Barat
5	Universitas Diponegoro	0.5623	Semarang	Jawa Tengah
6	Universitas Brawijaya	0.5612	Malang	Jawa Timur
7	Universitas Airlangga	0.5592	Surabaya	Jawa Timur
8	Institut Teknologi Sepuluh Nopember	0.5411	Surabaya	Jawa Timur
9	Universitas Muhammadiyah Malang	0.5279	Malang	Jawa Timur
10	Universitas Padjadjaran	0.5224	Bandung	Jawa Barat

Gambar 4.5 Antarmuka Daftar Peringkat Universitas

4.7.5 Antarmuka Rincian Universitas

Berikut ini merupakan tampilan antarmuka rincian universitas. Pengunjung dapat melihat rincian data dan nilai setiap universitas yang menjadi patokan perhitungan akhir dari daftar peringkat universitas seperti terlihat pada Gambar 4.6.



Gambar 4.6 Antarmuka Rincian Universitas

BAB V

PENGUJIAN DAN EVALUASI

Bab ini membahas hasil dan pembahasan pada aplikasi yang dikembangkan. Pada bab ini akan dijelaskan tentang data yang digunakan, hasil yang didapatkan dari penggunaan perangkat lunak dan uji coba yang dilakukan pada perangkat lunak yang telah dikerjakan untuk menguji apakah fungsionalitas aplikasi telah diimplementasikan dengan benar dan berjalan sebagaimana mestinya.

5.1 Lingkungan Uji Coba

Lingkungan uji coba menjelaskan lingkungan yang digunakan untuk menguji implementasi pembuatan sistem pada tugas akhir ini. Lingkungan uji coba meliputi perangkat keras dan perangkat lunak yang dijelaskan sebagai berikut:

1. Perangkat keras
 - a. Prosesor: Intel® Core™ i5-3230 @2.60GHz, RAM 4GB dengan Sistem Operasi Windows 10 Professional
 - b. Memori (RAM): 4 GB
 - c. Tipe sistem: Windows 10 Professional
2. Perangkat lunak
 - a. Sistem operasi: Windows 10 Professional
 - b. Perangkat pengembang: Python-2.7.11

5.2 Data Uji Coba

Data uji coba yang digunakan sebagai masukan adalah abstrak *paper* 16 universitas dengan peringkat teratas pada QS University Rankings. Setiap universitas mendapatkan abstrak *paper* sebanyak 50 buah dengan tema rumpun ilmu teknik. Total jumlah semua abstrak *paper* yakni 800 buah.

Untuk menguji kebenaran dari hasil uji coba, digunakan data *groundtruth* berupa peringkat universitas yang ada pada QS University Rankings.

5.2.1 Data Dokumen Abstrak Paper Akademisi

Data dokumen abstrak yang digunakan sebagai dataset berasal dari dokumen abstrak paper akademisi dari masing-masing universitas. Data-data tersebut diambil dari Google Scholar. Data yang diambil sebanyak 50 buah dari masing-masing 16 universitas tertinggi di Indonesia yang bersumber dari pemeringkatan QS.

Kata-kata yang terdapat pada abstrak akan diproses melalui prapemrosesan seperti *case folding*, tokenisasi, pembuangan tanda baca, dan *stopwords-removal*.

5.2.2 Data Dokumen Abstrak Paper Akademisi

Sebagai acuan corpus TKT, digunakan data corpus yang bernama Taxonomy Bloom. Taxonomy Bloom memiliki 6 (enam) level yang diisi oleh daftar kata atau *keyword* yang sesuai dengan ke-6 topik. Corpus Taxonomy Bloom yang digunakan berdasarkan atas beberapa sumber dari internet, yakni di antaranya:

- a. www.cte.cornell.edu
- b. www.bloomtaxonomy.org
- c. www.teaching.uncc.edu
- d. www.web.stanford.edu

Setelah pengumpulan corpus dari berbagai sumber, *keyword-keyword* tersebut digabungkan dan dikelompokkan menyesuaikan dengan masing-masing topik levelnya. Nantinya, corpus Taxonomy Bloom akan digunakan sebagai upaya untuk memperkaya corpus TKT yang akan dijadikan sebagai acuan. Berikut merupakan daftar 6 (enam) level Taxonomy Bloom seperti pada Tabel 5.2.

Tabel 5.1 Gabungan Jumlah Kata Taxonomy Bloom

No.	Level	Jumlah Kata
1.	Knowledge	35
2.	Comprehension	29
3.	Application	36

4.	Analysis	51
5.	Synthesis	51
6.	Evaluation	46

5.2.3 Data Jumlah Mahasiswa dan Tenaga Pengajar

Data jumlah mahasiswa yang digunakan mengacu pada data perguruan tinggi yang terdaftar pada situs PDDIKTI. Perguruan tinggi yang terdaftar pada situs PDDIKTI meliputi Institut, Universitas, Sekolah Tinggi, Sekolah Tinggi Ilmu Kesehatan, Akademi, dan Politeknik.

Data jumlah tenaga pengajar juga mempunyai kriteria khusus sebagai aturan ketika ingin memasukkan dataset ke sistem. Beberapa kriteria yang digunakan sebagai masukkan sistem dari atribut, di antaranya:

1. Berdasarkan status pendidikan tertinggi
 - a) Jenjang diploma (D1, D2, D3, D4)
 - b) Jenjang sarjana (S1, S2, S3)
 - c) Jenjang spesialis (SP1, SP2)
 - d) Jenjang profesi
 - e) Jenjang non-formal dan informal
2. Berdasarkan jabatan
 - a) Jabatan akademik lektor
 - b) Jabatan akademik lektor kepala
 - c) Jabatan akademik profesor
3. Berdasarkan status kepegawaian
 - a) Dosen dengan status tetap
 - b) Dosen dengan status tidak tetap

5.2.4 Data Online Science and Technology Index (SINTA)

Data online dari Science and Technology Index (SINTA) berupa data yang berhubungan dengan jumlah penelitian akademisi perguruan tinggi. Data yang disediakan meliputi Scopus yang tidak dapat secara gratis digunakan. Oleh karena itu,

dalam kriteria ini proses pengambilan data menggunakan teknik *grabbing* dari halaman situs SINTA.

Berikut beberapa konten yang dapat diambil dari situs SINTA dan digunakan dalam kriteria penilaian:

- 1) *Authors*, merupakan jumlah penulis penelitian yang ter-index oleh Google Scholar dan Scopus.
- 2) *Google Citation*, merupakan jumlah sitasi dari semua author dalam perguruan tinggi yang ter-index oleh Google Scholar.
- 3) *Google Documents*, merupakan jumlah dokumen penelitian (paper) yang ter-index Google Scholar.
- 4) *Google Scholar Indexed*, merupakan jumlah h-index dan i10-index dari setiap akademisi yang ter-index oleh Google Scholar.
- 5) *Scopus Indexed*, merupakan jumlah h-index dan i10-index dari setiap akademisi yang ter-index oleh Scopus.
- 6) *Journal Articles*, merupakan jumlah paper penelitian yang masuk dalam jurnal.
- 7) *Articles in Scopus*, merupakan artikel penelitian dari akademisi yang masuk dalam database Scopus.
- 8) *Non Articles in Scopus*, merupakan non artikel yang ter-index oleh database Scopus.

Sebagai catatan bahwa data yang disediakan oleh website SINTA untuk Tugas Akhir ini kurang valid dikarenakan basis data SINTA tidak diperbarui secara berkala. Data SINTA yang digunakan pada Tugas Akhir ini adalah data periode 2016-2017. Data ini perlu dilakukan validasi terhadap periode terbaru setiap tahunnya. Alasan menggunakan data dari SINTA ini adalah karena data yang disediakan mudah didapat dan sesuai dengan variabel indikator penilaian Ground Truth QS World University Rankings yaitu jumlah sitasi. Jika data yang digunakan pada SINTA diperbarui maka otomatis hasil pemeringkatan Perguruan Tinggi pada Tugas Akhir ini akan berubah juga sesuai kondisi data *real time* dengan bobot indikator sebesar 20%.

5.2.5 Data Statistik dan Analitik *Online*

Data statistic dan analitik *online* merupakan data yang berkuat seputar *traffic* sebuah web serta data-data statistic mengenai situs tersebut. Data universitas berserta statistiknya juga dapat dilihat pada situs-situs yang berkuat pada aspek ini.

Teknik *grabbing* dapat dilakukan sebagai upaya dalam mengumpulkan data seputar pemeringkatan universitas. Biasanya, ada beberapa situs yang menyediakan cara atau teknik dalam melakukan *grabbing*. Selain itu, banyak situs-situs pemeringkatan universitas di dunia yang menggunakan jasa-jasa perusahaan ini karena terpercaya dan mudah untuk digunakan.

Pengambilan data *online* ini memanfaatkan beberapa API tambahan yang tersebar gratis di internet untuk memaksimalkan hasil dan memudahkan analisis. API yang digunakan antara lain Alexa Rank API, dan MOZ Analytic API.

5.2.5.1 Alexa Rank

Alexa Rank API berfokus pada pengalihan informasi traffict rank dari *webhost* (domain, subdomain). Informasi yang didapatkan menggunakan Alexa Rank API antara lain:

1. *World Rank*, merupakan peringkat *webhost* dalam peringkat dunia. Peringkat ini sangat mempengaruhi kepopuleran sebuah situs. Semakin tinggi peringkat, maka akan semakin kecil nominal ranking yang didapatkan.
2. *Country Rank*, merupakan peringkat *webhost* dalam lingkup negara. Peringkat ini diperoleh dari analisis lokasi *webhost* berasal.
3. *Country Name*, merupakan informasi lokasi negara di mana *webhost* berasal.

5.2.5.2 MOZ Analytic

MOZ Analytic API berfokus pada penggalian informasi kepopuleran *webhost* yang dilihat dari jumlah *backlink* yang

dimiliki. Informasi yang didapatkan menggunakan MOZ Analytic API antara lain:

1. *Domain Authority* (DA), merupakan tingkat traffict pada domain utama maupun subdomain website perguruan tinggi.
2. *Page Authority* (PA), merupakan tingkat traffict pada page tertentu saja dari website perguruan tinggi.
3. *External Backlink*, jumlah *traffic* akses pengguna terhadap link yang terdapat pada sebuah halaman situs lain.
4. *Domain Name*, merupakan informasi domain name milik *webhost*.

5.3 Skenario Pengujian

Scenario pengujian berikut yang dilakukan dalam tugas akhir memiliki beberapa tahapan pengujian seperti pada subbab berikut ini.

5.3.1 Prapemrosesan

Dataset yang digunakan pada scenario pengujian ini yaitu sebanyak 800 dataset dari 16 peringkat universitas Indonesia tertinggi bersumber dari QS. Berikut merupakan rincian jumlah kata hasil dari sebelum dan sesudah proses prapemrosesan pada Tabel 5.2.

Tabel 5.2 Jumlah Kata Hasil Pemrosesan

No.	Perguruan Tinggi	Kata Asli	Kata Hasil Preprocess
1.	Institut Pertanian Bogor	10539	6293
2.	Institut Teknologi Bandung	9287	5675
3.	Institut Teknologi Sepuluh Nopember	9630	6015
4.	Universitas 17 Agustus 1945	9146	5580
5.	Univeritas Airlangga	9457	6009
6.	Universitas Bina Nusantara	7236	4432

7.	Universitas Brawijaya	8651	5469
8.	Universitas Diponegoro	8397	5327
9.	Universitas Gadjah Mada	10214	6279
10.	Universitas Indonesia	8823	5645
11.	Universitas Islam Indonesia	7897	4989
12.	Universitas Mataram	7527	4561
13.	Universitas Muhammadiyah Surakarta	10330	6117
14.	Universitas Padjadjaran	7774	5008
15.	Universitas Pelita Harapan	7204	4542
16.	Universitas Sebelas Maret	7808	5244

Seperti terlihat pada Tabel 5.2, hasil proses pemrosesan berhasil mengurangi kata yang tidak penting. Hasil ini berguna dalam memudahkan proses selanjutnya dengan jumlah fitur atau kata yang jauh lebih sedikit daripada sebelumnya.

5.3.2 Memperkaya Corpus TKT

Corpus TKT sebagai acuan berhasil diperluas hingga menjadi 9 level. Corpus TKT berhasil diluaskan dengan bantuan POS-Tagging. Hal ini dibantu dengan pencocokan sinonim atau persamaan kata pada setiap level TKT. Jika terjadi kecocokan, maka kata terjadi diambil dan dimasukkan ke dalam corpus level TKT yang bersangkutan. Berikut merupakan rincian corpus TKT final setelah terjadi pada Tabel 5.3 berikut.

Tabel 5.3 Hasil Akuisisi POS-Tagging 1

No.	Level	Jumlah kata default	Corpus final dengan POS Tagging Akuisisi Ke-1
1.	TKT 1	31	175
2.	TKT 2	24	118
3.	TKT 3	32	128
4.	TKT 4	15	79
5.	TKT 5	31	296
6.	TKT 6	24	214

7.	TKT 7	26	240
8.	TKT 8	30	142
9.	TKT 9	34	133
Total Keseluruhan		256	1525

5.3.3 Fitur Seleksi Corpus TKT

Pada pengujian ini, dijalankan penggunaan Helmholtz sebagai upaya menseleksi fitur pada corpus TKT yang tidak penting. Berikut ini merupakan hasil proses seleksi fitur yang dilakukan oleh Helmholtz terhadap corpus TKT seperti pada Tabel 5.4 berikut.

Tabel 5.4 Hasil Fitur Seleksi Helmholtz

No.	Level	Corpus final TKT	Corpus final hasil Helmholtz
1.	TKT 1	175	157
2.	TKT 2	118	102
3.	TKT 3	128	112
4.	TKT 4	79	68
5.	TKT 5	296	270
6.	TKT 6	214	195
7.	TKT 7	240	211
8.	TKT 8	142	120
9.	TKT 9	133	106
Total Keseluruhan		1525	1341

5.3.4 Pembobotan Metode Generatif Model L-LDA

Berikut ini merupakan hasil dari uji coba perhitungan *term* dengan menggunakan L-LDA. Perhitungan *term* akan menentukan tiga *term* tertinggi di setiap dokumen dari masing-masing universitas. Setelah itu, akan dilakukan proses menggunakan L-LDA untuk menghitung probabilitas dari masing-masing topik. Probabilitas tersebut kemudian dihitung

dengan menggunakan bobot sebagai nilai akhir untuk dijadikan peringkat setiap universitas. Berikut merupakan rincian hasil akhir reputasi akademik POS-Tagging setiap universitas seperti pada Tabel 5.5 beserta peringkat setiap universitas pada Tabel 5.6. Kemudian, pada Tabel 5.7 dan Tabel 5.8 merupakan rincian hasil akhir reputasi akademik WordNet level 1. Selanjutnya, pada Tabel 5.9 dan Tabel 5.10 merupakan rincian hasil akhir reputasi akademik WordNet level 2. Lalu, pada Tabel 5.11 dan Tabel 5.12 merupakan rincian hasil akhir reputasi akademik WordNet level 3. Terakhir, pada Tabel 5.13 dan Tabel 5.14 merupakan rincian hasil akhir reputasi akademik WordNet level all.

Tabel 5.5 Nilai Akhir Universitas POS-Tagging

No.	Universitas	Nilai Akhir
1.	BINUS	6.611
2.	IPB	6.612
3.	ITB	6.704
4.	ITS	6.645
5.	UGM	6.660
6.	UI	6.706
7.	UII	6.680
8.	UMS	6.648
9.	UNAIR	6.668
10.	UNBRAW	6.636
11.	UNDIP	6.699
12.	UNPAD	6.689
13.	UNRAM	6.622
14.	UNS	6.644
15.	UNTAG	6.655
16.	UPH	6.636

Tabel 5.6 Peringkat Universitas POS-Tagging

Nama	Peringkat Ground Truth	Peringkat Sistem POS-Tagging	Beda	Toleransi (≤ 3)
BINUS	13	11	2	1
IPB	6	4	2	1
ITB	1	1	0	1
ITS	9	6	3	1
UB	8	8	0	1
UGM	3	3	0	1
UI	2	2	0	1
UMS	11	10	1	1
UNAIR	4	9	5	
UNDIP	5	5	0	1
UNPAD	7	7	0	1
BINUS	10	11	1	1
TOTAL			18	11

Tabel 5.7 Nilai Akhir Universitas WordNet level 1

Nama	Peringkat Ground Truth	Peringkat Sistem WordNet level 1	Beda	Toleransi (≤ 3)
IPB	5	4	1	1
ITB	1	3	2	1
ITS	7	5	2	1
UB	9	7	2	1
UGM	2	1	1	1
UI	3	9	6	
UMS	8	2	6	
UNAIR	4	8	4	
UNDIP	6	6	0	1
TOTAL			24	5

Tabel 5.8 Peringkat Universitas WordNet level 1

Nama	Peringkat Ground Truth	Peringkat Sistem WordNet level 1	Beda	Toleransi (≤ 3)
IPB	5	4	1	1
ITB	2	1	1	1
UI	7	6	1	1
UNAIR	9	7	2	1
UMS	3	2	1	1
UGM	1	3	2	1
UNDIP	8	8	0	1
ITS	4	9	5	
UB	6	5	1	1
TOTAL			14	8

Tabel 5.9 Nilai Akhir Universitas WordNet level 2

Nama	Peringkat Ground Truth	Peringkat Sistem WordNet level 2	Beda	Toleransi (≤ 3)
IPB	5	5	0	1
ITB	1	2	1	1
ITS	7	3	4	
UB	9	6	3	1
UGM	2	9	7	
UI	3	1	2	1
UMS	8	8	0	1
UNAIR	4	7	3	1
UNDIP	6	4	2	1
TOTAL			22	7

Tabel 5.10 Peringkat Universitas WordNet level 2

Nama	Peringkat Ground Truth	Peringkat Sistem WordNet level 2	Beda	Toleransi (≤ 3)
IPB	5	4	1	1

ITB	2	1	1	1
UI	7	5	2	1
UNAIR	9	6	3	1
UMS	3	3	0	1
UGM	1	2	1	1
UNDIP	8	8	0	1
ITS	4	7	3	1
UB	6	9	3	1
TOTAL			18	9

Tabel 5.11 Nilai Akhir Universitas WordNet level 3

Nama	Peringkat Ground Truth	Peringkat Sistem WordNet level 2	Beda	Toleransi (≤ 3)
IPB	5	2	3	1
ITB	1	3	2	1
ITS	7	7	0	1
UB	9	8	1	1
UGM	2	4	2	1
UI	3	5	2	1
UMS	8	1	7	
UNAIR	4	9	5	
UNDIP	6	6	0	1
TOTAL			22	7

Tabel 5.12 Peringkat Universitas WordNet level 3

Nama	Peringkat Ground Truth	Peringkat Sistem WordNet level 3	Beda	Toleransi (≤ 3)
IPB	5	4	1	1
ITB	2	1	1	1
UI	7	6	1	1
UNAIR	9	7	2	1
UMS	3	3	0	1
UGM	1	2	1	1

UNDIP	8	8	0	1
ITS	4	9	5	
UB	6	5	1	1
TOTAL			18	8

Tabel 5.13 Nilai Akhir Universitas WordNet level all

Nama	Peringkat Ground Truth	Peringkat Sistem WordNet level 2	Beda	Toleransi (≤ 3)
IPB	5	1	4	
ITB	1	2	1	1
ITS	7	8	1	1
UB	9	9	0	1
UGM	2	6	4	
UI	3	3	0	1
UMS	8	5	3	1
UNAIR	4	4	0	1
UNDIP	6	7	1	1
TOTAL			14	7

Tabel 5.14 Peringkat Universitas WordNet All

Nama	Peringkat Ground Truth	Peringkat Sistem WordNet level all	Beda	Toleransi (≤ 3)
IPB	5	3	2	1
ITB	2	1	1	1
UI	7	7	0	1
UNAIR	9	9	0	1
UMS	3	4	1	1
UGM	1	2	1	1
UNDIP	8	8	0	1
ITS	4	5	1	1
UB	6	6	0	1
TOTAL			6	9

Berikut merupakan perbandingan hasil POS-Tagging pada Tabel 5.15, WordNet level 1 Tabel 5.16, WordNet level 2 pada Tabel 5.17, WordNet level 3 pada Tabel 5.18, dan WordNet level all pada Tabel 5.19 di bawah ini.

Tabel 5.15 Perbandingan Hasil POS-Tagging

Lingkup	Indikator	Percobaan
Indikator Reputasi Akademik	Beda (error)	52
	Toleransi	10/16
	Prosentase toleransi	62,5%
	Keterangan prosentase toleransi	62,5% peringkat masuk dalam toleransi
Sistem secara keseluruhan (all indicator)	Beda (error)	18
	Toleransi	10/11
	Prosentase toleransi	90,91%
	Keterangan prosentase toleransi	90,91% peringkat masuk dalam toleransi
	Similarity Pearson Correlation	77,27%

Tabel 5.16 Perbandingan Hasil WordNet Level 1

Lingkup	Indikator	Percobaan
Indikator Reputasi Akademik (L-LDA & Helmholtz)	Beda (error)	24
	Toleransi	6/9
	Prosentase toleransi	66,67%
	Keterangan prosentase toleransi	66,67% peringkat masuk dalam toleransi
Sistem secara keseluruhan (all indicator)	Beda (error)	14
	Toleransi	8/9
	Prosentase toleransi	88,89%
	Keterangan prosentase toleransi	88,89% peringkat masuk dalam toleransi
	Similarity Pearson Correlation	68,33%

Tabel 5.17 Perbandingan Hasil Wordnet Level 2

Lingkup	Indikator	Percobaan
Indikator Reputasi Akademik (L-LDA & Helmholtz)	Beda (error)	22
	Toleransi	7/9
	Prosentase toleransi	77,78%
	Keterangan prosentase toleransi	77,78% peringkat masuk dalam toleransi
Sistem secara keseluruhan (all indicator)	Beda (error)	14
	Toleransi	9/9
	Prosentase toleransi	100%
	Keterangan prosentase toleransi	100% peringkat masuk dalam toleransi
	Similarity Pearson Correlation	71,67%

Tabel 5.18 Perbandingan Hasil Wordnet Level 3

Lingkup	Indikator	Percobaan
Indikator Reputasi Akademik (L-LDA & Helmholtz)	Beda (error)	22
	Toleransi	7/9
	Prosentase toleransi	77,78%
	Keterangan prosentase toleransi	77,78% peringkat masuk dalam toleransi
Sistem secara keseluruhan (all indicator)	Beda (error)	12
	Toleransi	8/9
	Prosentase toleransi	88,89%
	Keterangan prosentase toleransi	88,89% peringkat masuk dalam toleransi
	Similarity Pearson Correlation	71,67%

Tabel 5.19 Perbandingan Hasil Wordnet Level All

Lingkup	Indikator	Percobaan
Indikator Reputasi Akademik (L-LDA & Helmholtz)	Beda (error)	14
	Toleransi	8/9
	Prosentase toleransi	88,89%
	Keterangan prosentase toleransi	88,89% peringkat masuk dalam toleransi

Sistem secara keseluruhan (all indicator)	Beda (error)	6
	Toleransi	9/9
	Prosentase toleransi	100%
	Keterangan prosentase toleransi	100% peringkat masuk dalam toleransi
	Similarity Pearson Correlation	93,33%

Pada hasil eksperimen di atas, terlihat pada Tabel 5.15 bahwa *Helmholtz* berhasil meningkatkan *gap* reputasi akademik dari 52 menjadi 18 setelah semua indikator juga dilibatkan dalam perhitungan peringkat universitas. Dan menghasilkan nilai similaritas sebesar 77,27% terhadap QS World University Rankings 2016-2017.

Pada hasil eksperimen di atas, terlihat pada Tabel 5.16 bahwa *Helmholtz* berhasil meningkatkan *gap* reputasi akademik dari 24 menjadi 14 setelah semua indikator juga dilibatkan dalam perhitungan peringkat universitas. Dan menghasilkan nilai similaritas sebesar 68,33% terhadap QS World University Rankings 2016-2017.

Pada hasil eksperimen di atas, terlihat pada Tabel 5.17 bahwa *Helmholtz* berhasil meningkatkan *gap* reputasi akademik dari 22 menjadi 14 setelah semua indikator juga dilibatkan dalam perhitungan peringkat universitas. Dan menghasilkan nilai similaritas sebesar 71,67% terhadap QS World University Rankings 2016-2017.

Pada hasil eksperimen di atas, terlihat pada Tabel 5.18 bahwa *Helmholtz* berhasil meningkatkan *gap* reputasi akademik dari 22 menjadi 12 setelah semua indikator juga dilibatkan dalam perhitungan peringkat universitas. Dan menghasilkan nilai similaritas sebesar 71,67% terhadap QS World University Rankings 2016-2017.

Terakhir, pada hasil eksperimen di atas, terlihat pada Tabel 5.19 bahwa *Helmholtz* berhasil meningkatkan *gap* reputasi akademik dari 14 menjadi 6 setelah semua indikator juga dilibatkan dalam perhitungan peringkat universitas. Dan

menghasilkan nilai similaritas sebesar 93,33% terhadap QS World University Rankings 2016-2017.

5.4 Evaluasi Pengujian

Berdasarkan hasil percobaan, semua skenario pengujian berhasil dan program berjalan dengan baik. Sehingga bisa ditarik disimpulkan bahwa fungsionalitas dari program telah bisa bekerja sesuai dengan yang diharapkan. Berikut merupakan rangkuman hasil pengujian fungsionalitas program yang berhasil dan bekerja sesuai dengan harapan.

ID	Nama	Hasil Uji Coba
UJ-UC-001	Memasukkan presentase bobot kriteria penilaian pemeringkatan	Berhasil
UJ-UC-002	Meihat peringkat perguruan tinggi berdasar tipe pemeringkatan	Berhasil
UJ-UC-003	Mencari perguruan tinggi berdasarkan potongan nama	Berhasil
UJ-UC-004	Mencari perguruan tinggi berdasarkan region wilayah provinsi	Berhasil
UJ-UC-005	Melihat detail informasi utama perguruan tinggi	Berhasil

[Halaman ini sengaja dikosongkan]

BAB VI

KESIMPULAN DAN SARAN

Pada bab ini akan diberikan kesimpulan yang diambil selama pengerjaan Tugas Akhir serta saran-saran tentang pengembangan yang dapat dilakukan terhadap Tugas Akhir ini di masa yang akan datang.

6.1 Kesimpulan

Dari hasil pengamatan selama proses perancangan, implementasi, dan pengujian perangkat lunak yang dilakukan, dapat diambil kesimpulan sebagai berikut:

1. Proses POS-Tagging dan WordNet memperkaya *corpus* TKT sehingga meningkatkan akurasi kecocokkan *keyword* yang ada pada *corpus* TKT.
2. Proses POS-Tagging dan WordNet yang jumlah katanya bertambah sangat membantu dalam mengurangi efek sampling duplikasi kata.
3. Proses penggunaan Helmholtz berhasil mendapatkan *gap* peringkat reputasi akademik universitas terkecil sebesar 14 poin yang dibantu dengan Wordnet.
4. Proses penggunaan Helmholtz berhasil mendapatkan persentase terbesar Similaritas Pearson Correlation sebesar 93.33% yang dibantu dengan WordNet.
5. Indikator-indikator lain pada saat penghitungan akhir peringkat universitas berhasil membantu menaikkan nilai reputasi akademik.
6. Penggunaan data berbasis SINTA dalam Tugas Akhir ini kurang valid, karena data yang digunakan tidak diperbaharui secara berkala oleh SINTA.

6.2 Saran

Berikut merupakan beberapa saran untuk pengembangan sistem di masa yang akan datang. Saran-saran ini didasarkan pada hasil perancangan, implementasi dan pengujian yang telah dilakukan.

1. Menambahkan dan/atau memperbaiki cara dalam memperkaya corpus dengan POS-Tagging atau sinonim word (WordNet) dapat meningkatkan koleksi *keyword* dalam corpus.
2. Proses *grabbing* memakan banyak waktu. Dapat digunakan cara lain yang sekiranya mengurangi waktu kerja tersebut.
3. Perlu adanya penambahan kriteria agar tidak hanya memperhatikan dari kriteria utama saja dan juga dapat dikondisikan agar seragam dalam kriteria penilaiannya.
4. Data kriteria jumlah sitasi dan kualitas sitasi serta data yang bersifat kuantitatif terhadap akademisi menggunakan data dari Scopus, karena data yang disediakan telah lengkap dan diperbaharui secara berkala.

DAFTAR PUSTAKA

- [1] K.S. Reddy, En Xie, Qingqing Tang, "Higher education, high-impact research, and world university rankings: A Case of India and comparison with China," *Pacific Science Review B: Humanities and Social Sciences* 2, pp. 1-21, 2016.
- [2] Tatiana Sidorenko, Tatiana Gorbatoeva, "Efficiency of Russian education through the scale of," *Procedia - Social and Behavioral Sciences* 166, pp. 464-467, 2015.
- [3] Wikipedia, "Wikipedia College and University Rankings," 14 December 2016. [Online]. Available: https://en.wikipedia.org/wiki/College_and_university_rankings. [Accessed 18 December 2016].
- [4] Adina - Petruta Pavel, "Global University Rankings - A Comparative Analysis," *Procedia Economics and Finance* 26, pp. 54-63, 2015.
- [5] 4ICU, [Online]. Available: <http://www.4icu.org/>. [Accessed 10 12 2016].
- [6] QS World University Rankings, "QS Top University," 1 January 2016. [Online]. Available: <http://www.topuniversities.com/university-rankings/asian-university-rankings/2016>. [Accessed 10 December 2016].
- [7] Mu-Hsuan Huang, "Opening The Black Box of QS World University Rankings," *Research Evaluation* 21, pp. 71-78, 2012.
- [8] QS World University Rankings, "QS Global Academic Survey 2014," QS World University Rankings, 2014.
- [9] Vidya Rajiv Yeravdekar, Gauri Tiwari, "Global Rankings of Higher Education Institutions and India's Effective Non-Precense: Why Have World-Class Universities Eluded The Indian Higher Education System? And, How Worthwhile is The Indian Government's Captivation to Launch World Class Universities," *Procedia - Social and Behavioral Sciences* 157, pp. 63-83, 2014.
- [10] QS World University Rankings, "Annual QS Global Employer Survey," QS World University Rankings, 2014.
- [11] L. R. Raymond J. Mooney, "Content-based book recommending

- using learning for text categorization," in *DL '00 Proceedings of the fifth ACM conference on Digital libraries*, San Antonio, Texas, USA, 2000.
- [12] Wikipedia, "Wikipedia Scopus," 4 August 2016. [Online]. Available: <https://en.wikipedia.org/wiki/Scopus>. [Accessed 10 December 2016].
 - [13] Elsevier, "Elsevier Developers," 1 January 2016. [Online]. Available: <http://dev.elsevier.com/>. [Accessed 10 December 2016].
 - [14] Anne-Wil K. Harzing, Ron van der Wal, "Google Scholar as A New Source for Citation Analysis," *Ethics in Science and Environmental Politics*, pp. 61-73, 2008.
 - [15] Wikipedia, "Wikipedia Google Scholar," 27 June 2017. [Online]. Available: https://en.wikipedia.org/wiki/Google_Scholar. [Accessed 17 December 2016].
 - [16] Wikipedia, "Wikipedia H-Index," 28 July 2017. [Online]. Available: <https://en.wikipedia.org/wiki/H-index>. [Accessed 28 July 2017].
 - [17] Kementerian Riset, Teknologi dan Pendidikan Tinggi, "Profil, Fitur, Registrasi Sinta Pusat Index, Sitasi dan Kepakaran Terbesar di Indonesia," Kementerian Riset, Teknologi dan Pendidikan Tinggi, Jakarta, 2017.
 - [18] Kementerian Riset, teknologi dan Pendidikan Tinggi, "Sinta Science and Technology Index," Kementerian Riset Teknologi dan Pendidikan Tinggi, 28 July 2017. [Online]. Available: <http://sinta1.ristekdikti.go.id/>. [Accessed 28 July 2017].
 - [19] Kementerian Riset Teknologi dan Pendidikan Tinggi Republik Indonesia, "Kementerian Riset Teknologi dan Pendidikan Tinggi Republik Indonesia," 1 January 2016. [Online]. Available: <http://www.forlap.dikti.go.id/>. [Accessed 10 December 2016].
 - [20] Arjun Thakur, A. L. Sangal, Harminder Bindra, "Quantitative Measurement and Comparison of Effects of Various SearchEngine Optimization Parameters on Alexa Traffic Rank," *International Journal of Computer Applications*, pp. 975-8887, 2011.
 - [21] Wikipedia, "Wikipedia Python Programming Language," 28 July 2017. [Online]. Available: [https://en.wikipedia.org/wiki/Python_\(programming_language\)](https://en.wikipedia.org/wiki/Python_(programming_language)). [Accessed 28 July 2017].

- [22] Steven Bird, "NLTK: The Natural Language Toolkit," *Proceedings of the COLINGACL on Interactive presentation*, pp. 69-72, 2006.
- [23] Tingting Wei, Yonghe Lu, Houyou Chang, Qiang Zhou, Xianyu Bao, "A Semantic Approach for Text Clustering using WordNet and Lexical Chains," *Expert Systems with Applications* 42, vol. 4, no. 1, pp. 2264-2275, 2015.
- [24] Fabian Pedregosa, Gael Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, "Scikit-learn: Machine Learning in Python," *Journal of Machine Learning Research* 12, pp. 2825-2830, 2011.
- [25] TextBlob, "TextBlob: Simplified Text Processing," 28 June 2017. [Online]. Available: <https://textblob.readthedocs.io/en/dev/>. [Accessed 28 June 2017].
- [26] Wikipedia, "Wikipedia PHP," 28 July 2017. [Online]. Available: <https://en.wikipedia.org/wiki/PHP>. [Accessed 28 July 2017].
- [27] Wikipedia, "Wikipedia MySQL," 28 July 2017. [Online]. Available: <https://en.wikipedia.org/wiki/MySQL>. [Accessed 28 July 2017].
- [28] Daniel Jurafsky, James H. Martin, "Part-of-Speech Tagging," in *Speech and Language Processing*, 2016.
- [29] Center of Teaching and Learning Stanford University, "Bloom's Taxonomy of Educational Objectives," Stanford University, California, 1981.
- [30] M. Tutkan, M. C. Ganiz and S. Akyoku, "Helmholtz principle based supervised and unsupervised feature selection methods for text mining," *Information Processing and Management*, p. 26, 2016.
- [31] A. Pradhan, "Helmholtz Principle-Based," National Institute of Technology Rourkela, Odisha, 2013.
- [32] William M. Darling, "A Theoretical and Practical Implementation Tutorial on Topic Modeling and Gibbs Sampling," 1 December 2011.
- [33] David M. Blei, Andrew Y. Ng, Michael I. Jordan, "Latent Dirichlet Allocation," *Journal of Machine Learning Research* 20, pp. 993-1022, 2003.
- [34] Andreyan Rizky Baskara, Riyanarto Sarno, Adhatus Solichah, "Discovering Traceability between Business Process and Software Component using Latent Dirichlet Allocation," 2016.

- [35] Daniel Ramage, David Hall, Ramesh Nallapati, Christopher D. Manning, "Labeled LDA: A supervised topic model for credit attribution in multi-labeled corpora," 2009.
- [36] Juan Ramos, "Using TF-IDF to Determine Word Relevance in Document Queries," 2003.
- [37] Badan Akreditasi Nasional Perguruan Tinggi, "Pedoman Penyusunan Borang Akreditasi Institusi Perguruan Tinggi," Badan Akreditasi Nasional Perguruan Tinggi, Jakarta, 2011.
- [38] Kementerian, Riset Teknologi dan Pendidikan Tinggi, "Peraturan Menteri Riset, Teknologi, dan Pendidikan Tinggi Republik Indonesia Nomor 42 Tahun 2016 Tentang Pengukuran dan Penetapan Tingkat Kesiapterapan Teknologi," Kementerian Riset, Teknologi dan Pendidikan Tinggi, Jakarta, 2016.

LAMPIRAN

```

1. #!/usr/bin/python
2. # -*- coding: latin-1 -*-
3.
4. import sys
5. from nltk.corpus import stopwords
6. from nltk.stem import PorterStemmer
7. from nltk.stem import SnowballStemmer
8. from nltk.tokenize import word_tokenize, sent_tokenize, RegexpTokenizer
9. import re, string, nltk, os, math, random, time
10. from textblob import TextBlob
11.
12. #fungsi sorting str+number (campuran) tipe karakter
13. def atoi(text):
14.     return int(text) if text.isdigit() else text
15.
16. def natural_keys(text):
17.     return [ atoi(c) for c in re.split('(\d+)', text) ]
18.
19. def perkaya_corpus_tkt(tmp_result, list_update_tkt_all):
20.     openfile_tkt1 = open("D:/TA/Helmholtz/Code/Ger/llda/percobaan/helmholtz_postagger/dokumen_tkt/A_Da
tatrain/TKT1.txt", "r")
21.     content_tkt1 = openfile_tkt1.read()
22.     content_tkt1 = content_tkt1.split()
23.     openfile_tkt1.close()
24.
25.     openfile_tkt2 = open("D:/TA/Helmholtz/Code/Ger/llda/percobaan/helmholtz_postagger/dokumen_tkt/A_Da
tatrain/TKT2.txt", "r")
26.     content_tkt2 = openfile_tkt2.read()
27.     content_tkt2 = content_tkt2.split()
28.     openfile_tkt2.close()
29.
30.     openfile_tkt3 = open("D:/TA/Helmholtz/Code/Ger/llda/percobaan/helmholtz_postagger/dokumen_tkt/A_Da
tatrain/TKT3.txt", "r")
31.     content_tkt3 = openfile_tkt3.read()

```

```
32.     content_tkt3 = content_tkt3.split()
33.     openfile_tkt3.close()
34.
35.     openfile_tkt4 = open("D:/TA/Helmholtz/Code/Ger/
    llda/percobaan/helmholtz_postagger/dokumen_tkt/A_Da
    tatrain/TKT4.txt", "r")
36.     content_tkt4 = openfile_tkt4.read()
37.     content_tkt4 = content_tkt4.split()
38.     openfile_tkt4.close()
39.
40.     openfile_tkt5 = open("D:/TA/Helmholtz/Code/Ger/
    llda/percobaan/helmholtz_postagger/dokumen_tkt/A_Da
    tatrain/TKT5.txt", "r")
41.     content_tkt5 = openfile_tkt5.read()
42.     content_tkt5 = content_tkt5.split()
43.     openfile_tkt5.close()
44.
45.     openfile_tkt6 = open("D:/TA/Helmholtz/Code/Ger/
    llda/percobaan/helmholtz_postagger/dokumen_tkt/A_Da
    tatrain/TKT6.txt", "r")
46.     content_tkt6 = openfile_tkt6.read()
47.     content_tkt6 = content_tkt6.split()
48.     openfile_tkt6.close()
49.
50.     openfile_tkt7 = open("D:/TA/Helmholtz/Code/Ger/
    llda/percobaan/helmholtz_postagger/dokumen_tkt/A_Da
    tatrain/TKT7.txt", "r")
51.     content_tkt7 = openfile_tkt7.read()
52.     content_tkt7 = content_tkt7.split()
53.     openfile_tkt7.close()
54.
55.     openfile_tkt8 = open("D:/TA/Helmholtz/Code/Ger/
    llda/percobaan/helmholtz_postagger/dokumen_tkt/A_Da
    tatrain/TKT8.txt", "r")
56.     content_tkt8 = openfile_tkt8.read()
57.     content_tkt8 = content_tkt8.split()
58.     openfile_tkt8.close()
59.
60.     openfile_tkt9 = open("D:/TA/Helmholtz/Code/Ger/
    llda/percobaan/helmholtz_postagger/dokumen_tkt/A_Da
    tatrain/TKT9.txt", "r")
61.     content_tkt9 = openfile_tkt9.read()
62.     content_tkt9 = content_tkt9.split()
```



```

63.     openfile_tkt9.close()
64.
65.     tmp_result = tmp_result.split()
66.     tmp_word = ""
67.     listWord = []
68.     lw = {}
69.     tmp_result_2 = ""
70.
71.     for word in tmp_result:
72.         if word in lw:
73.             lw[word] += 1;
74.         else:
75.             lw[word] = 1;
76.             tmp_result_2 += word + " "
77.
78.     listWord.append(lw)
79.     tmp_result_2_final = tmp_result_2.split()
80.
81.     #tambahan
82.     index=0
83.     for word in tmp_result_2_final:
84.
85.         if word in content_tkt1:
86.             text_postagger = nltk.word_tokenize(word)
87.             pos = nltk.pos_tag(text_postagger)
88.             list_update_tkt_all.append(word)
89.
90.             for i in range(0, len(pos)):
91.                 jos = pos[i][1]
92.                 if jos=='NN':
93.                     list_update_tkt_all.append(tmp_
result_2_final[index-1])
94.                     list_update_tkt_all.append(tmp_
result_2_final[index+1])
95.
96.                 elif jos=='NNS':
97.                     list_update_tkt_all.append(tmp_
result_2_final[index-1])
98.                     list_update_tkt_all.append(tmp_
result_2_final[index+1])
99.
100.                    elif jos=='NNP':

```

```

101.                list_update_tkt_all.append(tmp_result_2_final[index-1])
102.                list_update_tkt_all.append(tmp_result_2_final[index+1])
103.
104.                elif jos=='NNPS':
105.                    list_update_tkt_all.append(tmp_result_2_final[index-1])
106.                    list_update_tkt_all.append(tmp_result_2_final[index+1])
107.
108.                elif jos=='VBD':
109.                    list_update_tkt_all.append(tmp_result_2_final[index-1])
110.                    list_update_tkt_all.append(tmp_result_2_final[index+1])
111.
112.                elif jos=='VBG':
113.                    list_update_tkt_all.append(tmp_result_2_final[index-1])
114.                    list_update_tkt_all.append(tmp_result_2_final[index+1])
115.
116.                elif jos=='VBN':
117.                    list_update_tkt_all.append(tmp_result_2_final[index-1])
118.                    list_update_tkt_all.append(tmp_result_2_final[index+1])
119.
120.                elif jos=='VBP':
121.                    list_update_tkt_all.append(tmp_result_2_final[index-1])
122.                    list_update_tkt_all.append(tmp_result_2_final[index+1])
123.
124.                elif jos=='VBZ':
125.                    list_update_tkt_all.append(tmp_result_2_final[index-1])
126.                    list_update_tkt_all.append(tmp_result_2_final[index+1])
127.
128.            if word in content_tkt2:

```

```

129.         size(word)
130.             pos = nltk.pos_tag(text_postagger
131.                                 r)
132.                 list_update_tkt_all.append(word)
133.
134.         for i in range(0, len(pos)):
135.             jos = pos[i][1]
136.             if jos=='NN':
137.                 list_update_tkt_all.appe
138. nd(tmp_result_2_final[index-1])
139.                 list_update_tkt_all.appe
140. nd(tmp_result_2_final[index+1])
141.
142.             elif jos=='NNS':
143.                 list_update_tkt_all.appe
144. nd(tmp_result_2_final[index-1])
145.                 list_update_tkt_all.appe
146. nd(tmp_result_2_final[index+1])
147.
148.             elif jos=='NNP':
149.                 list_update_tkt_all.appe
150. nd(tmp_result_2_final[index-1])
151.                 list_update_tkt_all.appe
152. nd(tmp_result_2_final[index+1])
153.
154.             elif jos=='NNPS':
155.                 list_update_tkt_all.appe
156. nd(tmp_result_2_final[index-1])
157.                 list_update_tkt_all.appe
158. nd(tmp_result_2_final[index+1])
159.
160.             elif jos=='VBD':
161.                 list_update_tkt_all.appe
162. nd(tmp_result_2_final[index-1])
163.                 list_update_tkt_all.appe
164. nd(tmp_result_2_final[index+1])
165.
166.             elif jos=='VBG':
167.                 list_update_tkt_all.appe
168. nd(tmp_result_2_final[index-1])
169.                 list_update_tkt_all.appe
170. nd(tmp_result_2_final[index+1])

```

```

157.                 list_update_tkt_all.append(tmp_result_2_final[index+1])
158.
159.                 elif jos=='VBN':
160.                     list_update_tkt_all.append(tmp_result_2_final[index-1])
161.                     list_update_tkt_all.append(tmp_result_2_final[index+1])
162.
163.                 elif jos=='VBP':
164.                     list_update_tkt_all.append(tmp_result_2_final[index-1])
165.                     list_update_tkt_all.append(tmp_result_2_final[index+1])
166.
167.                 elif jos=='VBZ':
168.                     list_update_tkt_all.append(tmp_result_2_final[index-1])
169.                     list_update_tkt_all.append(tmp_result_2_final[index+1])
170.
171.                 if word in content_tkt3:
172.                     text_postagger = nltk.word_tokenize(word)
173.                     pos = nltk.pos_tag(text_postagger)
174.                     list_update_tkt_all.append(word)
175.
176.                 for i in range(0, len(pos)):
177.                     jos = pos[i][1]
178.                     if jos=='NN':
179.                         list_update_tkt_all.append(tmp_result_2_final[index-1])
180.                         list_update_tkt_all.append(tmp_result_2_final[index+1])
181.
182.                     elif jos=='NNS':
183.                         list_update_tkt_all.append(tmp_result_2_final[index-1])
184.                         list_update_tkt_all.append(tmp_result_2_final[index+1])
185.

```

```

186.                 elif jos=='NNP':
187.                     list_update_tkt_all.append(tmp_result_2_final[index-1])
188.                     list_update_tkt_all.append(tmp_result_2_final[index+1])
189.
190.                 elif jos=='NNPS':
191.                     list_update_tkt_all.append(tmp_result_2_final[index-1])
192.                     list_update_tkt_all.append(tmp_result_2_final[index+1])
193.
194.                 elif jos=='VBD':
195.                     list_update_tkt_all.append(tmp_result_2_final[index-1])
196.                     list_update_tkt_all.append(tmp_result_2_final[index+1])
197.
198.                 elif jos=='VBG':
199.                     list_update_tkt_all.append(tmp_result_2_final[index-1])
200.                     list_update_tkt_all.append(tmp_result_2_final[index+1])
201.
202.                 elif jos=='VBN':
203.                     list_update_tkt_all.append(tmp_result_2_final[index-1])
204.                     list_update_tkt_all.append(tmp_result_2_final[index+1])
205.
206.                 elif jos=='VBP':
207.                     list_update_tkt_all.append(tmp_result_2_final[index-1])
208.                     list_update_tkt_all.append(tmp_result_2_final[index+1])
209.
210.                 elif jos=='VBZ':
211.                     list_update_tkt_all.append(tmp_result_2_final[index-1])
212.                     list_update_tkt_all.append(tmp_result_2_final[index+1])
213.
214.             if word in content_tkt4:

```

```

215.                text_postagger = nltk.word_token
                ize(word)
216.                pos = nltk.pos_tag(text_postagge
                r)
217.                list_update_tkt_all.append(word)

218.
219.                for i in range(0, len(pos)):
220.                    jos = pos[i][1]
221.                    if jos=='NN':
222.                        list_update_tkt_all.appe
                nd(tmp_result_2_final[index-1])
223.                        list_update_tkt_all.appe
                nd(tmp_result_2_final[index+1])
224.
225.                    elif jos=='NNS':
226.                        list_update_tkt_all.appe
                nd(tmp_result_2_final[index-1])
227.                        list_update_tkt_all.appe
                nd(tmp_result_2_final[index+1])
228.
229.                    elif jos=='NNP':
230.                        list_update_tkt_all.appe
                nd(tmp_result_2_final[index-1])
231.                        list_update_tkt_all.appe
                nd(tmp_result_2_final[index+1])
232.
233.                    elif jos=='NNPS':
234.                        list_update_tkt_all.appe
                nd(tmp_result_2_final[index-1])
235.                        list_update_tkt_all.appe
                nd(tmp_result_2_final[index+1])
236.
237.                    elif jos=='VBD':
238.                        list_update_tkt_all.appe
                nd(tmp_result_2_final[index-1])
239.                        list_update_tkt_all.appe
                nd(tmp_result_2_final[index+1])
240.
241.                    elif jos=='VBG':
242.                        list_update_tkt_all.appe
                nd(tmp_result_2_final[index-1])

```

```

243.         list_update_tkt_all.append(tmp_result_2_final[index+1])
244.
245.         elif jos=='VBN':
246.             list_update_tkt_all.append(tmp_result_2_final[index-1])
247.             list_update_tkt_all.append(tmp_result_2_final[index+1])
248.
249.         elif jos=='VBP':
250.             list_update_tkt_all.append(tmp_result_2_final[index-1])
251.             list_update_tkt_all.append(tmp_result_2_final[index+1])
252.
253.         elif jos=='VBZ':
254.             list_update_tkt_all.append(tmp_result_2_final[index-1])
255.             list_update_tkt_all.append(tmp_result_2_final[index+1])
256.
257.         if word in content_tkt5:
258.             text_postagger = nltk.word_tokenize(word)
259.             pos = nltk.pos_tag(text_postagger)
260.             list_update_tkt_all.append(word)
261.
262.         for i in range(0, len(pos)):
263.             jos = pos[i][1]
264.             if jos=='NN':
265.                 list_update_tkt_all.append(tmp_result_2_final[index-1])
266.                 list_update_tkt_all.append(tmp_result_2_final[index+1])
267.
268.             elif jos=='NNS':
269.                 list_update_tkt_all.append(tmp_result_2_final[index-1])
270.                 list_update_tkt_all.append(tmp_result_2_final[index+1])
271.

```

```

272.                     elif jos=='NNP':
273.                         list_update_tkt_all.append(tmp_result_2_final[index-1])
274.                         list_update_tkt_all.append(tmp_result_2_final[index+1])
275.
276.                     elif jos=='NNPS':
277.                         list_update_tkt_all.append(tmp_result_2_final[index-1])
278.                         list_update_tkt_all.append(tmp_result_2_final[index+1])
279.
280.                     elif jos=='VBD':
281.                         list_update_tkt_all.append(tmp_result_2_final[index-1])
282.                         list_update_tkt_all.append(tmp_result_2_final[index+1])
283.
284.                     elif jos=='VBG':
285.                         list_update_tkt_all.append(tmp_result_2_final[index-1])
286.                         list_update_tkt_all.append(tmp_result_2_final[index+1])
287.
288.                     elif jos=='VBN':
289.                         list_update_tkt_all.append(tmp_result_2_final[index-1])
290.                         list_update_tkt_all.append(tmp_result_2_final[index+1])
291.
292.                     elif jos=='VBP':
293.                         list_update_tkt_all.append(tmp_result_2_final[index-1])
294.                         list_update_tkt_all.append(tmp_result_2_final[index+1])
295.
296.                     elif jos=='VBZ':
297.                         list_update_tkt_all.append(tmp_result_2_final[index-1])
298.                         list_update_tkt_all.append(tmp_result_2_final[index+1])
299.
300.                 if word in content_tkt6:

```



```

301.                text_postagger = nltk.word_token
                 size(word)
302.                pos = nltk.pos_tag(text_postagge
                 r)
303.                list_update_tkt_all.append(word)

304.
305.                for i in range(0, len(pos)):
306.                    jos = pos[i][1]
307.                    if jos=='NN':
308.                        list_update_tkt_all.appe
nd(tmp_result_2_final[index-1])
309.                        list_update_tkt_all.appe
nd(tmp_result_2_final[index+1])
310.
311.                    elif jos=='NNS':
312.                        list_update_tkt_all.appe
nd(tmp_result_2_final[index-1])
313.                        list_update_tkt_all.appe
nd(tmp_result_2_final[index+1])
314.
315.                    elif jos=='NNP':
316.                        list_update_tkt_all.appe
nd(tmp_result_2_final[index-1])
317.                        list_update_tkt_all.appe
nd(tmp_result_2_final[index+1])
318.
319.                    elif jos=='NNPS':
320.                        list_update_tkt_all.appe
nd(tmp_result_2_final[index-1])
321.                        list_update_tkt_all.appe
nd(tmp_result_2_final[index+1])
322.
323.                    elif jos=='VBD':
324.                        list_update_tkt_all.appe
nd(tmp_result_2_final[index-1])
325.                        list_update_tkt_all.appe
nd(tmp_result_2_final[index+1])
326.
327.                    elif jos=='VBG':
328.                        list_update_tkt_all.appe
nd(tmp_result_2_final[index-1])

```

```

329.                 list_update_tkt_all.append(tmp_result_2_final[index+1])
330.
331.                 elif jos=='VBN':
332.                     list_update_tkt_all.append(tmp_result_2_final[index-1])
333.                     list_update_tkt_all.append(tmp_result_2_final[index+1])
334.
335.                 elif jos=='VBP':
336.                     list_update_tkt_all.append(tmp_result_2_final[index-1])
337.                     list_update_tkt_all.append(tmp_result_2_final[index+1])
338.
339.                 elif jos=='VBZ':
340.                     list_update_tkt_all.append(tmp_result_2_final[index-1])
341.                     list_update_tkt_all.append(tmp_result_2_final[index+1])
342.
343.                 if word in content_tkt7:
344.                     text_postagger = nltk.word_tokenize(word)
345.                     pos = nltk.pos_tag(text_postagger)
346.                     list_update_tkt_all.append(word)
347.
348.                 for i in range(0, len(pos)):
349.                     jos = pos[i][1]
350.                     if jos=='NN':
351.                         list_update_tkt_all.append(tmp_result_2_final[index-1])
352.                         list_update_tkt_all.append(tmp_result_2_final[index+1])
353.
354.                     elif jos=='NNS':
355.                         list_update_tkt_all.append(tmp_result_2_final[index-1])
356.                         list_update_tkt_all.append(tmp_result_2_final[index+1])
357.

```

```

358.                 elif jos=='NNP':
359.                     list_update_tkt_all.append(tmp_result_2_final[index-1])
360.                     list_update_tkt_all.append(tmp_result_2_final[index+1])
361.
362.                 elif jos=='NNPS':
363.                     list_update_tkt_all.append(tmp_result_2_final[index-1])
364.                     list_update_tkt_all.append(tmp_result_2_final[index+1])
365.
366.                 elif jos=='VBD':
367.                     list_update_tkt_all.append(tmp_result_2_final[index-1])
368.                     list_update_tkt_all.append(tmp_result_2_final[index+1])
369.
370.                 elif jos=='VBG':
371.                     list_update_tkt_all.append(tmp_result_2_final[index-1])
372.                     list_update_tkt_all.append(tmp_result_2_final[index+1])
373.
374.                 elif jos=='VBN':
375.                     list_update_tkt_all.append(tmp_result_2_final[index-1])
376.                     list_update_tkt_all.append(tmp_result_2_final[index+1])
377.
378.                 elif jos=='VBP':
379.                     list_update_tkt_all.append(tmp_result_2_final[index-1])
380.                     list_update_tkt_all.append(tmp_result_2_final[index+1])
381.
382.                 elif jos=='VBZ':
383.                     list_update_tkt_all.append(tmp_result_2_final[index-1])
384.                     list_update_tkt_all.append(tmp_result_2_final[index+1])
385.
386.             if word in content_tkt8:

```

```

387.                text_postagger = nltk.word_token
           ize(word)
388.                pos = nltk.pos_tag(text_postagge
           r)
389.                list_update_tkt_all.append(word)

390.
391.                for i in range(0, len(pos)):
392.                    jos = pos[i][1]
393.                    if jos=='NN':
394.                        list_update_tkt_all.appe
nd(tmp_result_2_final[index-1])
395.                        list_update_tkt_all.appe
nd(tmp_result_2_final[index+1])
396.
397.                    elif jos=='NNS':
398.                        list_update_tkt_all.appe
nd(tmp_result_2_final[index-1])
399.                        list_update_tkt_all.appe
nd(tmp_result_2_final[index+1])
400.
401.                    elif jos=='NNP':
402.                        list_update_tkt_all.appe
nd(tmp_result_2_final[index-1])
403.                        list_update_tkt_all.appe
nd(tmp_result_2_final[index+1])
404.
405.                    elif jos=='NNPS':
406.                        list_update_tkt_all.appe
nd(tmp_result_2_final[index-1])
407.                        list_update_tkt_all.appe
nd(tmp_result_2_final[index+1])
408.
409.                    elif jos=='VBD':
410.                        list_update_tkt_all.appe
nd(tmp_result_2_final[index-1])
411.                        list_update_tkt_all.appe
nd(tmp_result_2_final[index+1])
412.
413.                    elif jos=='VBG':
414.                        list_update_tkt_all.appe
nd(tmp_result_2_final[index-1])

```

```

415.         list_update_tkt_all.append(tmp_result_2_final[index+1])
416.
417.         elif jos=='VBN':
418.             list_update_tkt_all.append(tmp_result_2_final[index-1])
419.             list_update_tkt_all.append(tmp_result_2_final[index+1])
420.
421.         elif jos=='VBP':
422.             list_update_tkt_all.append(tmp_result_2_final[index-1])
423.             list_update_tkt_all.append(tmp_result_2_final[index+1])
424.
425.         elif jos=='VBZ':
426.             list_update_tkt_all.append(tmp_result_2_final[index-1])
427.             list_update_tkt_all.append(tmp_result_2_final[index+1])
428.
429.         if word in content_tkt9:
430.             text_postagger = nltk.word_tokenize(word)
431.             pos = nltk.pos_tag(text_postagger)
432.             list_update_tkt_all.append(word)
433.
434.         for i in range(0, len(pos)):
435.             jos = pos[i][1]
436.             if jos=='NN':
437.                 list_update_tkt_all.append(tmp_result_2_final[index-1])
438.                 list_update_tkt_all.append(tmp_result_2_final[index+1])
439.
440.             elif jos=='NNS':
441.                 list_update_tkt_all.append(tmp_result_2_final[index-1])
442.                 list_update_tkt_all.append(tmp_result_2_final[index+1])
443.

```

```

444.             elif jos=='NNP':
445.                 list_update_tkt_all.append(tmp_result_2_final[index-1])
446.                 list_update_tkt_all.append(tmp_result_2_final[index+1])
447.
448.             elif jos=='NNPS':
449.                 list_update_tkt_all.append(tmp_result_2_final[index-1])
450.                 list_update_tkt_all.append(tmp_result_2_final[index+1])
451.
452.             elif jos=='VBD':
453.                 list_update_tkt_all.append(tmp_result_2_final[index-1])
454.                 list_update_tkt_all.append(tmp_result_2_final[index+1])
455.
456.             elif jos=='VBG':
457.                 list_update_tkt_all.append(tmp_result_2_final[index-1])
458.                 list_update_tkt_all.append(tmp_result_2_final[index+1])
459.
460.             elif jos=='VBN':
461.                 list_update_tkt_all.append(tmp_result_2_final[index-1])
462.                 list_update_tkt_all.append(tmp_result_2_final[index+1])
463.
464.             elif jos=='VBP':
465.                 list_update_tkt_all.append(tmp_result_2_final[index-1])
466.                 list_update_tkt_all.append(tmp_result_2_final[index+1])
467.
468.             elif jos=='VBZ':
469.                 list_update_tkt_all.append(tmp_result_2_final[index-1])
470.                 list_update_tkt_all.append(tmp_result_2_final[index+1])
471.
472.             index+=1

```

```

473.
474.     #menentukan asumsi label awal untuk inputan
    komputasi L-LDA
475.     #fungsi untuk menentukan label awal mengguna
    kan keywprd bloom taxonomy (fix)
476.     def set_label(tmp_result):
477.         sys.stderr.write("JUMLAH KATA DALAM DOKU
    MEN: "+str(len(tmp_result))+"\n")
478.
479.         openfile_tkt1 = open("D:/TA/Helmholtz/Co
    de/Ger/llda/percobaan/helmholtz_postagger/dokumen_t
    kt/A_Datatrain_Update/update_TKT1.txt", "r")
480.         content_tkt1 = openfile_tkt1.read()
481.         openfile_tkt1.close()
482.
483.         openfile_tkt2 = open("D:/TA/Helmholtz/Co
    de/Ger/llda/percobaan/helmholtz_postagger/dokumen_t
    kt/A_Datatrain_Update/update_TKT2.txt", "r")
484.         content_tkt2 = openfile_tkt2.read()
485.         openfile_tkt2.close()
486.
487.         openfile_tkt3 = open("D:/TA/Helmholtz/Co
    de/Ger/llda/percobaan/helmholtz_postagger/dokumen_t
    kt/A_Datatrain_Update/update_TKT3.txt", "r")
488.         content_tkt3 = openfile_tkt3.read()
489.         openfile_tkt3.close()
490.
491.         openfile_tkt4 = open("D:/TA/Helmholtz/Co
    de/Ger/llda/percobaan/helmholtz_postagger/dokumen_t
    kt/A_Datatrain_Update/update_TKT4.txt", "r")
492.         content_tkt4 = openfile_tkt4.read()
493.         openfile_tkt4.close()
494.
495.         openfile_tkt5 = open("D:/TA/Helmholtz/Co
    de/Ger/llda/percobaan/helmholtz_postagger/dokumen_t
    kt/A_Datatrain_Update/update_TKT5.txt", "r")
496.         content_tkt5 = openfile_tkt5.read()
497.         openfile_tkt5.close()
498.
499.         openfile_tkt6 = open("D:/TA/Helmholtz/Co
    de/Ger/llda/percobaan/helmholtz_postagger/dokumen_t
    kt/A_Datatrain_Update/update_TKT6.txt", "r")
500.         content_tkt6 = openfile_tkt6.read()

```

```

501.         openfile_tkt6.close()
502.
503.         openfile_tkt7 = open("D:/TA/Helmholtz/Co
de/Ger/llda/percobaan/helmholtz_postagger/dokumen_t
kt/A_Datatrain_Update/update_TKT7.txt", "r")
504.         content_tkt7 = openfile_tkt7.read()
505.         openfile_tkt7.close()
506.
507.         openfile_tkt8 = open("D:/TA/Helmholtz/Co
de/Ger/llda/percobaan/helmholtz_postagger/dokumen_t
kt/A_Datatrain_Update/update_TKT8.txt", "r")
508.         content_tkt8 = openfile_tkt8.read()
509.         openfile_tkt8.close()
510.
511.         openfile_tkt9 = open("D:/TA/Helmholtz/Co
de/Ger/llda/percobaan/helmholtz_postagger/dokumen_t
kt/A_Datatrain_Update/update_TKT9.txt", "r")
512.         content_tkt9 = openfile_tkt9.read()
513.         openfile_tkt9.close()
514.
515.
516.         #UPDATE
517.         openfile_tkt_all = open("D:/TA/Helmholtz
/Code/Ger/llda/percobaan/helmholtz_postagger/dokume
n_tkt/A_Datatrain_All/tampung_TKT.txt", "r")
518.         content_tkt_all = openfile_tkt_all.read(
)
519.         openfile_tkt_all.close()
520.
521.         counter1=counter2=counter3=counter4=coun
ter5=counter6=counter7=counter8=counter9=counter_al
l=0
522.
523.         tmp_result = tmp_result.split()
524.         tmp_word = ""
525.
526.         #hitung frekuensi kemunculan kata
527.         listWord = []
528.         lw = {}
529.         tmp_result_2 = ""
530.
531.         for word in tmp_result:
532.             if word in lw:

```



```

533.             lw[word] += 1;
534.         else:
535.             lw[word] = 1;
536.             tmp_result_2 += word + " "
537.
538.         listWord.append(lw)
539.
540.         tmp_result_2_final = tmp_result_2.split(
541. )
542.         for word in tmp_result_2_final:
543.             if word in content_tkt1:
544.                 counter1 += 1
545.
546.             if word in content_tkt2:
547.                 counter2 += 1
548.
549.             if word in content_tkt3:
550.                 counter3 += 1
551.
552.             if word in content_tkt4:
553.                 counter4 += 1
554.
555.             if word in content_tkt5:
556.                 counter5 += 1
557.
558.             if word in content_tkt6:
559.                 counter6 += 1
560.
561.             if word in content_tkt7:
562.                 counter7 += 1
563.
564.             if word in content_tkt8:
565.                 counter8 += 1
566.
567.             if word in content_tkt9:
568.                 counter9 += 1
569.
570.
571.         list_label = [counter1, counter2, counte
r3, counter4, counter5, counter6, counter7, counter
8, counter9]

```

```

572.         list_nama_label = ["T1", "T2", "T3", "T4", "
    T5", "T6", "T7", "T8", "T9"]
573.
574.         tmp_lnl = {}
575.         a=0
576.         for lnl in list_nama_label:
577.             tmp_lnl[a] = lnl
578.             a+=1
579.
580.         tmp_lb = {}
581.         b=0
582.         p = ""
583.         simpan = []
584.         for lb in list_label:
585.             if lb>0:
586.                 tmp_lb[b] = lb
587.                 simpan.append(str(tmp_lb[b])+"-
    "+str(tmp_lnl[b]))
588.                 p += tmp_lnl[b] + ","
589.
590.                 b+=1
591.
592.         a = sorted(simpan, key=natural_keys, rev
    erse=True)
593.         sys.stderr.write("SORTING HASIL TF (penc
    ocokan berdasarkan term): "+str(a)+"\n")
594.
595.
596.         ambil_4_label_teratas = a[:4]
597.         ambil_4_label_teratas = ' '.join(ambil_4
    _label_teratas)
598.
599.         b = ambil_4_label_teratas.split()
600.
601.         c = {}
602.         d=0
603.         e=0
604.         f = {}
605.         label_final = ""
606.         for j in b:
607.             d+=1
608.
609.             c[d] = j[-2:]

```

```

610.
611.         for k in b:
612.             label_final += str(c[d-e])+", "
613.             e+=1
614.
615.             label_final = label_final[:-1]
616.
617.             label_final_akhir = "["+label_final+"]"
618.
619.             label_final_akhir = re.sub('-',
620.             ', ', label_final_akhir)
621.
622.             return label_final_akhir
623.
624.         #fungsi untuk preprocessing teks bhs inggris
625.         menggunakan snowball stemmer, nltk stopwords, remov
626.         e punctuation (fix)
627.         def inggris(dir3, nama_universitas, dir2_inp
628.         ut, lala2, listTF, tipe_bahasa, list_update_tkt1, l
629.         ist_update_tkt2, list_update_tkt3, list_update_tkt4
630.         , list_update_tkt5, list_update_tkt6, list_update_t
631.         kt7, list_update_tkt8, list_update_tkt9, list_updat
632.         e_tkt_all):
633.             tipe_bahasa = " - TIPE INGGRIS"
634.             nama_file = "Dokumen_"
635.
636.             # 1 tokenisasi & remove punctuation
637.             file_input = dir2_input.lower().split()
638.
639.
640.             punctuation = re.compile(r'[^\\w.]')
641.             word_list_punctuation = [punctuation.sub
642.             (" ", word) for word in file_input]
643.             tmp_word_list_punctuation = [x.strip(' '
644.             ) for x in word_list_punctuation]
645.
646.             punctuation2 = re.compile(r'[-
647.             .?!,:;()|0-9]')
648.             word_list_punctuation2 = [punctuation2.s
649.             ub("", word) for word in tmp_word_list_punctuation]

```

```

637.         tmp_word_list_punctuation2 = [x.strip('
        ') for x in word_list_punctuation2]
638.
639.         # 2 stopwords & stemming
640.         snowball_stemmer = SnowballStemmer("engl
        ish")
641.         stop_words = {}
642.         stop_words = set(stopwords.words("englis
        h"))
643.
644.
645.         words_list = []
646.         for i in tmp_word_list_punctuation2:
647.             if i not in stop_words:
648.                 words_list.append(i)
649.
650.         listResult = []
651.         for word in words_list:
652.             listResult.append(word)
653.
654.         tf = {}
655.         for word in listResult:
656.             if word in tf:
657.                 tf[word] += 1;
658.             else:
659.                 tf[word] = 1;
660.
661.         listTF.append(tf)
662.
663.         tmp_result = ""
664.
665.         for word in listResult:
666.             if len(word) > 2:
667.                 tmp_result += word + " "
668.
669.         jumlah_kata_hasil_preprocessing = tmp_re
        sult.split()
670.
671.         dir3.write("JUMLAH KATA HASIL PREPROCESS
        ING: "+str(len(jumlah_kata_hasil_preprocessing))+"\
        n\n")
672.

```

```

673.         path_full_doc = "D:/TA/Helmholtz/Code/Ge
r/llda/percobaan/helmholtz_postagger/result_preproc
essing_dan_labeling_bloom/"+nama_universitas+"/amat
_final_full_result_preprocessing_"
674.         nama_file_full_doc_result = path_full_do
c+lala2+".txt"
675.         final_full = open(nama_file_full_doc_res
ult, 'w')
676.
677.         final_full.write(tmp_result)
678.         final_full.close()
#===== print hasil prep
rocessing (bersih) dari satu file dokumen yg isinya
banyak =====
679.
680.         #UPDATE
681.         perkaya_corpus_tkt(tmp_result, list_upda
te_tkt_all)
682.
683.         #UPDATE
684.         path_update_tkt_all = "D:/TA/Helmholtz/C
ode/Ger/llda/percobaan/helmholtz_postagger/dokumen_
tk/A_Datatrain_All/tampung_TKT_"+nama_universitas+
".txt"
685.         final_tkt_all = open(path_update_tkt_all
, 'w')
686.
687.         for i in range(0, len(list_update_tkt_al
l)):
688.             final_tkt_all.write(list_update_tkt_
all[i]+"\\n") # ===== wr
ite kata2 hasil perkaya corpus ditaruh dalam satu f
ile baru bernama tampung =====
689.
690.         final_tkt_all.close()
691.
692.         sys.stderr.write("-- Not Important!!! --
\\n")
693.         sys.stderr.write("SUKSES PREPROCESS!!!\\
n\\n")
694.
695.         #fungsi untuk translate dokumen bhs indo ke
bhs inggris (fix)

```

```

696.         def translate_to_english_function(inputan_file, nFiles, listTF):
697.             isi_file = inputan_file
698.             jumlah_file = nFiles
699.             list_tf = listTF
700.             tipe_bahasa = " - TIPE INDONESIA"
701.
702.             en_blob = TextBlob(isi_file)
703.             hasil_translate = en_blob.translate(to='en')
704.             sys.stderr.write("HASIL TRANSLATE:"+tipe_bahasa+"\n"+str(hasil_translate)+"\n\n")
705.
706.             inggris(hasil_translate, jumlah_file, list_tf, tipe_bahasa)
707.
708.         #fungsi persiapan awal (fix)
709.         def compute():
710.             nama_universitas = "Institut Teknologi Sepuluh Nopember"
711.             path = "D:/TA/Helmholtz/Code/Ger/llda/percobaan/helmholtz_postagger/dokumen_universitas/"+nama_universitas+"/"
712.             dirs = os.listdir(path)
713.             lala = path[80:]
714.             lala2 = lala[:-1]
715.             print lala2
716.
717.             #===== read all dokumen universitas =====
718.
719.             #tambahan
720.             path_full_doc = "D:/TA/Helmholtz/Code/Ger/llda/percobaan/helmholtz_postagger/result_preprocessing_dan_labeling_bloom/"+nama_universitas+"/full_result_preprocessing_"
721.             ekstensi_full_doc = ".txt"
722.             nama_file_full_doc_result = path_full_doc + lala2 + ekstensi_full_doc
723.             final = open(nama_file_full_doc_result, 'w')
724.             #===== read all dokumen universitas kemudian di jadikan satu file (belum bersih) =====

```

```

723.         nFiles = 0
724.         listTF = []
725.         tipe_bahasa = ""
726.
727.         list_update_tkt1 = []
728.         list_update_tkt2 = []
729.         list_update_tkt3 = []
730.         list_update_tkt4 = []
731.         list_update_tkt5 = []
732.         list_update_tkt6 = []
733.         list_update_tkt7 = []
734.         list_update_tkt8 = []
735.         list_update_tkt9 = []
736.
737.         #UPDATE
738.         list_update_tkt_all = []
739.
740.         for eachFile in dirs:
741.             nFiles += 1
742.             with open(path+eachFile, 'r+') as my
File:
743.                 inputan_file = myFile.read().rep
lace('\n', ' ')
744.                 final.write(inputan_file+" ")
745.             final.close()
746.
747.             dir2 = open("D:/TA/Helmholtz/Code/Ger/11
da/percobaan/helmholtz_postagger/result_preprocessi
ng_dan_labeling_bloom/"+nama_universitas+"/full_res
ult_preprocessing_"+lala2+".txt", "r")
748.             dir2_input = dir2.read()
#===== read dokumen yg full (be
lum bersih) untuk di preprocessing =====
749.
750.             dir3 = open("D:/TA/Helmholtz/Code/Ger/11
da/percobaan/helmholtz_postagger/result_preprocessi
ng_dan_labeling_bloom/"+nama_universitas+"/analisa_
hasil_preprocessing_"+lala2+".txt", "w")
751.             jumlah_kata = dir2_input.split()
752.             sys.stderr.write("JUMLAH SEMUA KATA: "+s
tr(len(jumlah_kata))+ "\n")
753.             dir3.write("JUMLAH SEMUA KATA: "+str(len
(jumlah_kata))+ "\n\n")

```

```

754.
755.         inggris(dir3, nama_universitas, dir2_inp
ut, lala2, listTF, tipe_bahasa, list_update_tkt1, l
ist_update_tkt2, list_update_tkt3, list_update_tkt4
, list_update_tkt5, list_update_tkt6, list_update_t
kt7, list_update_tkt8, list_update_tkt9, list_updat
e_tkt_all)
756.
757.         dir3.close()
758.
759.         #MAIN FUNCTION#
760.         if __name__ == "__main__":
761.             start_time = time.time()
762.
763.             compute()
764.
765.             time_akhir = time.time() - start_time
766.             sys.stderr.write("---
"+str(time_akhir)+" detik ---")

```

Kode Sumber 25 Preprocessing & Labeling Labeling

```

1. # encoding=utf8
2.
3. from __future__ import division, unicode_literals
4. import math
5. import re
6. import collections
7. from textblob import TextBlob as tb
8. import sys, os
9. from collections import Counter
10. import sys
11.
12. reload(sys)
13. sys.setdefaultencoding('utf8')
14.
15. #main function
16. nama_universitas = "Institut Pertanian Bogor"
17. path = "D:/TA/Helmholtz/Code/Ger/helmholtz/result_p
reprocess/"+nama_universitas+"/"
18. dirs = os.listdir(path)
19. nFiles = 0

```



```

20. bloblist = {}
21. bloblist2 = []
22. tmp_namafile = []
23.
24. i=0
25. for eachFile in dirs:
26.     nFiles += 1
27.     with open(path+eachFile, 'r+') as myFile:
28.
29.         inputan_file = myFile.read().replace('\n',
            ' ')
30.         bloblist[i] = inputan_file.split()
31.         bloblist2.append(bloblist[i])
32.
33.         i+=1
34.
35. array_L = []
36. array_B = []
37.
38. for i in range(0, len(bloblist2)):                #it
    erasi dokumen
39.     array_B.append(bloblist2[i])
40.
41.     j=0
42.     for j in range(0, len(bloblist2[i])):          #it
        erasi kata dalam dokumenh
43.         array_L.append(bloblist2[i][j])
44.
45.         j+=1
46.
47. print "\n"
48. print "NILAI L (Keseluruhan):", len(array_L)
49. print array_L
50.
51.
52. print "\n"
53. for i in range(0, len(array_B)):
54.     print "NILAI B (Masing-
        masing)", i, ":", len(array_B[i])
55.     print array_B[i]
56.
57. array_K = {}
58. for i in range(0, len(array_B)):

```

```

59.     for word in array_B[i]:
60.         # print word
61.         if word in array_K:
62.             array_K[word] += 1
63.         else:
64.             array_K[word] = 1
65.
66.     print "\n"
67.     print "NILAI K (Keseluruhan):", len(array_K)
68.     print array_K
69.
70.     array_M = []
71.     array_M2 = []
72.
73.     print "\n"
74.     print "NILAI M (Masing-masing):"
75.
76.     for i in range(0, len(bloblist2)):
77.         print ">> NILAI I ", i
78.         array_M.append(bloblist2[i])
79.
80.         array_M3 = {}
81.
82.         kurang = len(bloblist2[i])-1
83.
84.         j=0
85.         for j in range(0, len(bloblist2[i])-kurang):
86.             for word3 in bloblist2[i]:
87.                 if word3 in array_M3:
88.                     array_M3[word3] += 1
89.                 else:
90.                     array_M3[word3] = 1
91.             j += 1
92.
93.         print array_M3
94.         array_M2.append(array_M3)
95.
96.     array_N = []
97.     print "\n"
98.     for i in range(0, len(array_B)):
99.         print "NILAI N (Masing-
           masing)", i, ":", len(array_L)/len(array_B[i])

```

```

100.         array_N.append(len(array_L)/len(array_B[
101.             i]))
102.         print "\n"
103.
104.         po_key = []
105.         po_value_K = []
106.         for key, value_K in array_K.iteritems():
107.             po_key.append(key)
108.             po_value_K.append(value_K)
109.
110.         #tambahan
111.         path_full_doc_all = "D:/TA/Helmholtz/Code/Ge
112.             r/helmholtz/hasil_all2017/"+nama_universitas+"/"
113.         dirs = os.listdir(path)
114.         ekstensi_full_doc_all = ".txt"
115.         nama_file_full_doc_result_all = path_full_do
116.             c_all+"hasil_all"+ekstensi_full_doc_all
117.         final_all = open(nama_file_full_doc_result_a
118.             ll, 'w')
119.
120.         nama_file_full_doc_result_all_final = path_f
121.             ull_doc_all+"hasil_all_final"+ekstensi_full_doc_all
122.
123.         final_all_final = open(nama_file_full_doc_re
124.             sult_all_final, 'w')
125.
126.         tampung_list_kata_all = []
127.         # tampung_list_kata_all_final = []
128.
129.         for i in range(0, len(array_M2)):
130.             print "DOKUMEN KE-", i+1
131.             print "=====
132.                 ====="
133.
134.         tampung_list_kata = []
135.
136.         nama_file = str(i+1)
137.
138.         #tambahan
139.         path_full_doc = "D:/TA/Helmholtz/Code/Ge
140.             r/helmholtz/hasil_all2017/"+nama_universitas+"/"
141.         dirs = os.listdir(path)

```

```

134.         ekstensi_full_doc = ".txt"
135.         nama_file_full_doc_result = path_full_doc
136.         final = open(nama_file_full_doc_result,
137.         'w')
138.         for key2, value_K2 in array_M2[i].iteritems():
139.             if key2 in po_key:
140.                 haha = po_value_K[po_key.index(key2)]
141.
142.                 rumus = (float) (math.factorial(haha)/(math.factorial(haha -
143.                 value_K2) * math.factorial(value_K2)) * 1/(math.pow(array_N[i], value_K2-1)))
144.                 rumus2 = (float) (-
145.                 1)*(1/value_K2)*math.log(rumus)
146.                 tampung_list_kata_all.append(str(key2)+" : "+str(rumus2))
147.
148.                 if rumus2 > 0.5:
149.                     print "KATA:", key2
150.                     print "JUMLAH KATA per dokumen:", value_K2
151.                     print "JUMLAH KATA seluruh dokumen:", po_value_K[po_key.index(key2)]
152.                     print "Nilai N: :", array_N[i]
153.                     print "NFA:", rumus, "\n"
154.                     print "Meaningfulness:", rumus2, "\n"
155.
156.                 tampung_list_kata.append(key2)
157.
158.         print tampung_list_kata
159.         for j in range(0, len(tampung_list_kata)):
160.             final.write(tampung_list_kata[j]+"\\n")
161.
162.         final_all_final.write(tampung_list_kata[j]+" ")

```

```

160.
161.         final.close()
162.
163.         print "\n"
164.
165.         final_all_final.close()
166.
167.         #print all with frequency to one document
168.         for k in range(0, len(tampung_list_kata_all)
169.             ):
170.             final_all.write(tampung_list_kata_all[k]
171.                 +"\n")
172.         final_all.close()

```

Kode Sumber 26 Helmholtz

```

1.  #!/usr/bin/python
2.  # -*- coding: latin-1 -*-
3.
4.  import sys
5.  from nltk.corpus import stopwords
6.  from nltk.stem import PorterStemmer
7.  from nltk.stem import SnowballStemmer
8.  from nltk.tokenize import word_tokenize, sent_tokenize, RegexpTokenizer
9.  import re, string, nltk, os, math, random, time
10. from textblob import TextBlob
11.
12. #fungsi sorting str+number (campuran) tipe karakter
13. def atoi(text):
14.     return int(text) if text.isdigit() else text
15.
16. def natural_keys(text):
17.     return [ atoi(c) for c in re.split('(\d+)', text) ]
18.
19. def term_frequency(tmp_result, isi_dokumen, dir3, dir4):
20.

```

```
21.     openfile_tkt1 = open("D:/TA/Helmholtz/Code/Ger/
    helmholtz/dokumen_tkt/B_Level_All -
    Copy/amat_final_update_TKT1.txt", "r")
22.     content_tkt1 = openfile_tkt1.read()
23.     content_tkt1 = content_tkt1.split()
24.     openfile_tkt1.close()
25.
26.     openfile_tkt2 = open("D:/TA/Helmholtz/Code/Ger/
    helmholtz/dokumen_tkt/B_Level_All -
    Copy/amat_final_update_TKT2.txt", "r")
27.     content_tkt2 = openfile_tkt2.read()
28.     content_tkt2 = content_tkt2.split()
29.     openfile_tkt2.close()
30.
31.     openfile_tkt3 = open("D:/TA/Helmholtz/Code/Ger/
    helmholtz/dokumen_tkt/B_Level_All -
    Copy/amat_final_update_TKT3.txt", "r")
32.     content_tkt3 = openfile_tkt3.read()
33.     content_tkt3 = content_tkt3.split()
34.     openfile_tkt3.close()
35.
36.     openfile_tkt4 = open("D:/TA/Helmholtz/Code/Ger/
    helmholtz/dokumen_tkt/B_Level_All -
    Copy/amat_final_update_TKT4.txt", "r")
37.     content_tkt4 = openfile_tkt4.read()
38.     content_tkt4 = content_tkt4.split()
39.     openfile_tkt4.close()
40.
41.     openfile_tkt5 = open("D:/TA/Helmholtz/Code/Ger/
    helmholtz/dokumen_tkt/B_Level_All -
    Copy/amat_final_update_TKT5.txt", "r")
42.     content_tkt5 = openfile_tkt5.read()
43.     content_tkt5 = content_tkt5.split()
44.     openfile_tkt5.close()
45.
46.     openfile_tkt6 = open("D:/TA/Helmholtz/Code/Ger/
    helmholtz/dokumen_tkt/B_Level_All -
    Copy/amat_final_update_TKT6.txt", "r")
47.     content_tkt6 = openfile_tkt6.read()
48.     content_tkt6 = content_tkt6.split()
49.     openfile_tkt6.close()
50.
```

```

51.     openfile_tkt7 = open("D:/TA/Helmholtz/Code/Ger/
    helmholtz/dokumen_tkt/B_Level_All -
    Copy/amat_final_update_TKT7.txt", "r")
52.     content_tkt7 = openfile_tkt7.read()
53.     content_tkt7 = content_tkt7.split()
54.     openfile_tkt7.close()
55.
56.     openfile_tkt8 = open("D:/TA/Helmholtz/Code/Ger/
    helmholtz/dokumen_tkt/B_Level_All -
    Copy/amat_final_update_TKT8.txt", "r")
57.     content_tkt8 = openfile_tkt8.read()
58.     content_tkt8 = content_tkt8.split()
59.     openfile_tkt8.close()
60.
61.     openfile_tkt9 = open("D:/TA/Helmholtz/Code/Ger/
    helmholtz/dokumen_tkt/B_Level_All -
    Copy/amat_final_update_TKT9.txt", "r")
62.     content_tkt9 = openfile_tkt9.read()
63.     content_tkt9 = content_tkt9.split()
64.     openfile_tkt9.close()
65.
66.     counter1=counter2=counter3=counter4=counter5=co
    unter6=counter7=counter8=counter9=counter_all=0
67.     counter_not_in=0
68.
69.     tmp_result = tmp_result.split()
70.     tmp_word = ""
71.
72.
73.     #==== show and print ====
74.     dir4.write("JUMLAH KATA DALAM DOKUMEN: "+str(le
    n(tmp_result))+ "\n\n")
75.     #==== show and print ====
76.
77.
78.     #hitung frekuensi kemunculan kata
79.     listWord = []
80.     lw = {}
81.     tmp_result_2 = ""
82.
83.     for word in tmp_result:
84.         if word in lw:
85.             lw[word] += 1;

```

```

86.         # tmp_result_2 += word + " "
87.     else:
88.         lw[word] = 1;
89.         tmp_result_2 += word + " "
90.
91.     listWord.append(lw)
92.
93.
94.     #==== show and print =====
95.     dir4.write("JUMLAH KATA SINGLE : "+str(len(lw))
96. +"\n\n")
97.     #==== show and print =====
98.     tmp_result_counter1 = ""
99.     tmp_result_counter2 = ""
100.     tmp_result_counter3 = ""
101.     tmp_result_counter4 = ""
102.     tmp_result_counter5 = ""
103.     tmp_result_counter6 = ""
104.     tmp_result_counter7 = ""
105.     tmp_result_counter8 = ""
106.     tmp_result_counter9 = ""
107.
108.     tmp_result_counter_not_in = ""
109.     tmp_result_counter_in = ""
110.
111.     tmp_result_2_final = tmp_result_2.split(
112. )
113.     for word in tmp_result_2_final:
114.         if word in content_tkt1:
115.             counter1 += 1
116.             tmp_result_counter1 += word + "
117.
118.         elif word in content_tkt2:
119.             counter2 += 1
120.             tmp_result_counter2 += word + "
121.
122.         elif word in content_tkt3:
123.             counter3 += 1

```



```

124.                tmp_result_counter3 += word + "
125.
126.                elif word in content_tkt4:
127.                    counter4 += 1
128.                    tmp_result_counter4 += word + "
129.
130.                elif word in content_tkt5:
131.                    counter5 += 1
132.                    tmp_result_counter5 += word + "
133.
134.                elif word in content_tkt6:
135.                    counter6 += 1
136.                    tmp_result_counter6 += word + "
137.
138.                elif word in content_tkt7:
139.                    counter7 += 1
140.                    tmp_result_counter7 += word + "
141.
142.                elif word in content_tkt8:
143.                    counter8 += 1
144.                    tmp_result_counter8 += word + "
145.
146.                elif word in content_tkt9:
147.                    counter9 += 1
148.                    tmp_result_counter9 += word + "
149.
150.                elif word not in content_tkt1 and word not in content_tkt2 and word not in content_tkt3
151.                    and word not in content_tkt4 and word not in content_tkt5 and word not in content_tkt6 and word not in content_tkt7 and word not in content_tkt8 and word not in content_tkt9:
152.                    counter_not_in +=1
153.                    tmp_result_counter_not_in += word + " "

```

```

154.         else:
155.             tmp_result_counter_in += word +
156.             " "
157.         print "TKT 1:\n", tmp_result_counter1
158.
159.         print "TKT 2:\n", tmp_result_counter2
160.
161.         print "TKT 3:\n", tmp_result_counter3
162.
163.         print "TKT 4:\n", tmp_result_counter4
164.
165.         print "TKT 5:\n", tmp_result_counter5
166.
167.         print "TKT 6:\n", tmp_result_counter6
168.
169.         print "TKT 7:\n", tmp_result_counter7
170.
171.         print "TKT 8:\n", tmp_result_counter8
172.
173.         print "TKT 9:\n", tmp_result_counter9
174.
175.         list_label = [counter1, counter2, counter3, counter4, counter5, counter6, counter7, counter8, counter9]
176.         list_nama_label = ["T1", "T2", "T3", "T4", "T5", "T6", "T7", "T8", "T9"]
177.
178.         #==== show and print ====
179.         dir4.write("JUMLAH KATA NOT IN DOKUMEN:
180.         "+str(counter_not_in)+"\n\n")
181.         #==== show and print ====
182.
183.         #==== show and print ====
184.         dir4.write("FREKUENSI TERM:\n"+str(list_label)+"\n"+str(list_nama_label)+"\n\n")
185.         #==== show and print ====
186.
187.         tmp_lnl = {}
188.         a=0
189.         for lnl in list_nama_label:
190.             tmp_lnl[a] = lnl
191.             a+=1

```

```

191.
192.         tmp_lb = {}
193.         b=0
194.         p = ""
195.         simpan = []
196.         for lb in list_label:
197.             if lb>0:
198.                 tmp_lb[b] = lb
199.                 simpan.append(str(tmp_lb[b])+"-
"+str(tmp_lnl[b]))
200.                 p += tmp_lnl[b] + ","
201.
202.                 b+=1
203.
204.         a = sorted(simpan, key=natural_keys, reverse=True)
205.
206.
207.         #==== show and print ====
208.         sys.stderr.write("SORTING HASIL TF (penc
ocokan berdasarkan term):\n"+str(a)+"\n")
209.         dir4.write("SORTING HASIL TF (pencocokan
berdasarkan term):\n"+str(a)+"\n\n")
210.         #==== show and print ====
211.
212.         ambil_4_label_teratas = a[:3]
213.         ambil_4_label_teratas = ' '.join(ambil_4
_label_teratas)
214.
215.         b = ambil_4_label_teratas.split()
216.
217.         c = {}
218.         d=0
219.         e=0
220.         f = {}
221.         label_final = ""
222.         for j in b:
223.             d+=1
224.             # c[d] = j[2:]
225.             c[d] = j[-2:]
226.
227.         for k in b:
228.             label_final += str(c[d-e])+","

```

```

229.             e+=1
230.
231.             label_final = label_final[:-1]
232.             label_final_akhir = "["+label_final+"]"
233.
234.             label_final_akhir = re.sub('-',
235.             ', ', label_final_akhir)
236.
237.             # return label_final_akhir
238.             sys.stderr.write("SORTING HASIL TF AMBIL
239.             4 TERATAS:\n"+str(label_final_akhir)+"\n")
240.
241.             dir3.write(str(label_final_akhir)+" "+str
242.             r(isi_dokumen))
243.
244.             #fungsi persiapan awal (fix)
245.             def compute():
246.                 nama_universitas = "Institut Pertanian_B
247.                 ogor"
248.
249.                 dir2 = open("D:/TA/Helmholtz/Code/Ger/11
250.                 da/-
251.                 tanpa pos tagging/result_preprocessing_dan_labelin
252.                 g_bloom - helmholtz -
253.                 all2017/"+nama_universitas+"/amat_final_full_resul
254.                 t_preprocessing_"+nama_universitas+".txt", "r")
255.                 dir2_input = dir2.read()
256.                 isi_dokumen = dir2_input
257.
258.                 dir2.close()
259.
260.                 dir3 = open("D:/TA/Helmholtz/Code/Ger/11
261.                 da/-
262.                 tanpa pos tagging/result_term_frequency_dan_labeli
263.                 ng_bloom - helmholtz -
264.                 all2017/"+nama_universitas+"/label_dokumen_based_o
265.                 n_term_frequency_"+nama_universitas+".txt", "w")
266.                 dir4 = open("D:/TA/Helmholtz/Code/Ger/11
267.                 da/-
268.                 tanpa pos tagging/result_term_frequency_dan_labeli
269.                 ng_bloom - helmholtz -
270.                 all2017/"+nama_universitas+"/analisa_hasil_term_fr
271.                 equency_"+nama_universitas+".txt", "w")
272.

```

```

252.         term_frequency(dir2_input, isi_dokumen,
253.             dir3, dir4)
254.         dir3.close()
255.         dir4.close()
256.
257.         #MAIN FUNCTION#
258.         if __name__ == "__main__":
259.             start_time = time.time()
260.
261.             compute()
262.
263.             time_akhir = time.time() - start_time
264.             sys.stderr.write("---
265.                 "+str(time_akhir)+" detik ---")

```

Kode Sumber 27 Term Frequency

```

1.  #LLDA: Labeled Latent Dirichlet Allocation#
2.
3.  from optparse import OptionParser
4.  import sys, re, numpy
5.  import time
6.  import csv
7.
8.  def load_corpus(filename):
9.      nama_file = str(filename)
10.
11.      corpus = []
12.      labels = []
13.      labelmap = dict()
14.
15.      open_file_corpus = open(filename, 'r')
16.
17.      line = ""
18.
19.      for line in open_file_corpus:
20.          get_label_per_line = re.match(r'\[(.+?)\](.
21.              +)', line)
22.
23.          if get_label_per_line:

```

```

23.             label = get_label_per_line.group(1).split(
24.                 it(',')
25.                 for i in label:
26.                     labelmap[i] = 1
27.             line = get_label_per_line.group(2)
28.
29.         else:
30.             label = None
31.
32.         doc = re.findall(r'\w+(?:\s*\w+)?', line.lower())
33.
34.         if len(doc)>0:
35.             corpus.append(doc)
36.             labels.append(label)
37.
38.     open_file_corpus.close()
39.
40.     return labelmap.keys(), corpus, labels, name_file
41.
42.
43. class LLDA:
44.     def __init__(self, K, alpha, beta):
45.         self.alpha = alpha
46.         self.beta = beta
47.
48.     def complement_label(self, label):
49.         if not label:
50.             return numpy.ones(len(self.labelmap))
51.
52.         vec_complement_label = numpy.zeros(len(self.labelmap))
53.
54.         for i in label:
55.             vec_complement_label[self.labelmap[i]] =
56.                 1.0
57.
58.         return vec_complement_label
59.
60.     def term_to_id(self, word):
61.         if word not in self.vocas_id:

```

```

61.         voca_id = len(self.vocas)
62.         self.vocas_id[word] = voca_id
63.         self.vocas.append(word)
64.     else:
65.         voca_id = self.vocas_id[word]
66.
67.     return voca_id
68.
69.     def set_corpus(self, labelset, corpus, labels):
70.         #labelset.insert(0, "common")
71.         self.labelmap = dict(zip(labelset, range(len(labelset))))
72.         self.K = len(self.labelmap)
73.         self.vocas = []
74.         self.vocas_id = dict()
75.         self.labels = numpy.array([self.complement_
label(label) for label in labels])
76.         self.docs = [[self.term_to_id(word) for word
in doc] for doc in corpus]
77.
78.         M = len(corpus)
79.         V = len(self.vocas)
80.
81.         self.zmn = []
82.         self.nmz = numpy.zeros((M, self.K), dtype=int)
83.         self.nzt = numpy.zeros((self.K, V), dtype=int)
84.         self.nz = numpy.zeros(self.K, dtype=int)
85.
86.         i=1
87.         for m, doc, label in zip(range(M), self.docs, self.labels):
88.             nm = len(doc)
89.             zn = [numpy.random.multinomial(1, label
/label.sum()).argmax() for x in range(nm)] #a
mbil random value dari batas setiap label yang akan
digunakan untuk menjadi label awal setiap kata
90.             self.zmn.append(zn)
91.
92.             for t, z in zip(doc, zn):
93.                 self.nmz[m, z] += 1

```

```

94.         self.nzt[z, t] += 1
95.         self.nz[z] += 1
96.
97.         i+=1
98.
99.     def gibbs_sampling_inference(self):
100.         V = len(self.vocas)
101.
102.         for m, doc, label in zip(range(len(self.docs)), self.docs, self.labels):
103.             len_doc = len(doc)
104.             for n in range(len_doc):
105.                 t = doc[n]
106.                 z = self.zmn[m][n]
107.
108.                 self.nmz[m, z] -= 1
109.                 self.nzt[z, t] -= 1
110.                 self.nz[z] -= 1
111.
112.                 denom_alpha = self.nmz[m].sum() + self.K * self.alpha #hitung matriks cdt
                                     (probabilitas topik dengan dokumen)
113.                 denom_beta = self.nzt.sum(axis=1) + V * self.beta #hitung matriks cwt
                                     (probabilitas topik dengan kata)
114.
115.                 probabilitas_z = label * (self.nzt[:, t] + self.beta) / denom_beta * (self.nmz[m] + self.alpha) / denom_alpha
116.                 new_z = numpy.random.multinomial(1, probabilitas_z / probabilitas_z.sum()).argmax() #nentukan nilai random value di antara label yang didapatkan, nantinya akan digunakan sebagai kesimpulan label pada iterasi ke i
117.
118.                 self.zmn[m][n] = new_z
119.                 self.nmz[m, new_z] += 1
120.                 self.nzt[new_z, t] += 1
121.                 self.nz[new_z] += 1
122.
123.     def phi(self):
124.         V = len(self.vocas)

```



```

125.         nilai_phi = (self.nzt + self.beta) /
            (self.nz[:, numpy.newaxis] + V * self.beta)
126.
127.         return nilai_phi
128.
129.         def theta(self):
130.             n_alpha = self.nmz + self.labels * s
            elf.alpha
131.             nilai_theta = n_alpha / n_alpha.sum(
                axis=1)[:, numpy.newaxis]
132.
133.             return nilai_theta
134.
135.             #perplexity: sebuah pengukuran trhdhp seb
            erapa baik distribusi yang dijalankan dalam mempred
            iksi dan melakukan komputasi sampel. semakin rendah
            nilai perplexity maka semakin bagus distribusi tsb
            dalam memprediksi
136.         def perplexity(self, docs=None):
137.             if docs == None:
138.                 docs = self.docs
139.
140.                 nilai_phi = self.phi()
141.                 nilai_thetas = self.theta()
142.
143.                 N = 0
144.                 log_per = 0
145.
146.                 for doc, theta in zip(docs, nilai_th
                    etas):
147.                     for word in doc:
148.                         log_per -
                            = numpy.log(numpy.inner(nilai_phi[:, word], theta))
149.
150.                         N += len(doc)
151.                         result_perplexity = numpy.exp(lo
                            g_per/N)
152.                         print result_perplexity
153.
154.                         #nilai perplexity: pada iterasi
                            i nilainya jauh lebih besar daripada iterasi ke n+1
                            , hal ini dikarenakan sistem akan mempelajari dahul

```

```

    u pada iterasi ke i, dan setelah itu pada iterasi i
    +1 maka sistem akan belajar dari kegagalan di itera
    si ke i
155.
156.         return result_perplexity
157.
158.     def main():
159.         print "SEMANGAT"
160.
161.         parser = OptionParser()
162.         parser.add_option("-
f", dest="filename", help="corpus filename")    #ge
t nama file corpusnya
163.         parser.add_option("--
alpha", dest="alpha", type="float", help="parameter
alpha (parameter penyemibang jumlah topik T)", def
ault=25.0)
164.         parser.add_option("--
beta", dest="beta", type="float", help="parameter b
eta (parameter penyemibang jumlah kata W)", default
=0.01)
165.         parser.add_option("-
k", dest="K", type="int", help="jumlah topik utama
(sesuai jumlah TKT)", default=9)
166.         parser.add_option("-
i", dest="iteration", type="int", help="jumlah iter
asi gibbs sampling", default=100)
167.
168.         (options, args) = parser.parse_args()
169.         if not options.filename:
170.             parser.error("Masukkan nama file cor
pus anda (diawali dengan -f)")
171.
172.         labelset, corpus, labels, nama_file = lo
ad_corpus(options.filename)
173.
174.         #menampilkan labelset dan nama file#
175.         print labelset
176.         #print nama_file
177.
178.         lala = nama_file[130:]
179.         print lala
180.         lala2 = lala[:-4]

```

```

181.         print lala2
182.
183.         lala_final = lala2.replace("_", " ")
184.         print lala_final
185.         #selesai menampilkan labelset dan nama f
         ile#
186.
187.
188.         llda = LLDA(options.K, options.alpha, op
         tions.beta)
189.         llda.set_corpus(labelset, corpus, labels
         )
190.
191.         for i in range(options.iteration):
192.             llda.gibbs_sampling_inference()
193.             print '\n'
194.
195.             simpan_file = open("D:/TA/Helmholtz/Code
         /Ger/llda/-
         tanpa pos tagging/result_llda_with_tf_idf_sklearn
         - helmholtz -
         all2017/result_llda_"+lala_final+".txt", "w")
196.             simpan_file_hasil_theta = open("D:/TA/He
         lmholtz/Code/Ger/llda/-
         tanpa pos tagging/result_llda_with_tf_idf_sklearn
         - helmholtz -
         all2017/result_theta_max_"+lala_final+".txt", "w")
197.             simpan_file_hasil_theta_all = open("D:/T
         A/Helmholtz/Code/Ger/llda/-
         tanpa pos tagging/result_llda_with_tf_idf_sklearn
         - helmholtz -
         all2017/result_theta_all_"+lala_final+".txt", "w")
198.
199.             out = csv.writer(open("myfile_"+lala_fin
         al+".csv", "wb"), delimiter=',', quoting=csv.QUOTE_
         ALL)
200.
201.             phi = llda.phi()
202.             theta = llda.theta()
203.

```

```

204.         simpan_file.write(str(labelset))

        #simpan txt
205.         simpan_file.write("\n")

        #simpan txt
206.
207.         header_phi = '\n\n=====
===== PROBABILITAS PHI =====
===== '
208.
209.         simpan_file.write(header_phi+"\n")

        #simpan txt
210.
211.         for v, vocabulary in enumerate(llda.voca
s):
212.             tmp_hasil_label = ', '.join([vocabul
ary] + [str(hasil_x) for hasil_x in llda.nzt[:, v]]
)
213.             tmp_hasil_phi = ", ".join([vocabular
y] + [str(hasil_x) for hasil_x in phi[:, v]])
214.
215.             simpan_file.write(tmp_hasil_label)

        #simpan txt
216.         simpan_file.write("\n")

        #simpan txt
217.         simpan_file.write(tmp_hasil_phi)

        #simpan txt
218.         simpan_file.write("\n\n")

        #simpan txt
219.
220.         simpan_file.write("\n\n"+str(labelset))

        #simpan txt
221.
222.         header_theta = '\n\n\n=====
===== PROBABILITAS THETA =====
===== '

```

```

223.         print header_theta
224.         simpan_file.write(header_theta+"\n")

        #simpan txt

225.
226.         d=0
227.         tmp_posisi_labelmap_key_akhir = llda.K -
        1
228.         tmp_posisi_labelmap_key_awal = 0
229.
230.         for p, label_per_document in enumerate(1
        lda.labels):
231.             j=0
232.             flag=0
233.
234.             simpan = []
235.             simpan2 = {}
236.
237.             p+=1
238.             print "LABEL DOKUMEN", p, ":", label
        _per_document
239.             simpan_file.write("LABEL DOKUMEN "+s
        tr(p)+" "+str(label_per_document)+"\n")
        #simpan txt

240.
241.             for hasil_y in theta[d, :]:
242.                 if tmp_posisi_labelmap_key_awal
        > tmp_posisi_labelmap_key_akhir:
243.                     tmp_posisi_labelmap_key_awal
        = 0
244.
245.                     simpan.append(str(hasil_y))
246.                     simpan2[j] = hasil_y
247.                     print "Posisi Label ke -
        ", j, ":", simpan2[j]
248.
249.                     simpan_file_hasil_theta_all.writ
        e(str(labelset[j])+" "+str(simpan2[j])+"\n")
        #NEW_NEW

250.
251.             j+=1
252.

```

```

253.                tmp_posisi_labelmap_key_awal+=1
254.
255.                print "--- NILAI N ---:", j, "\n"
256.                d+=1
257.
258.                print "HASIL ARRAY PROBABILITAS\n",
simpan, "\n"
259.                simpan_file.write("HASIL TANPA SORTI
NG PROBABILITAS: "+str(simpan)+"\n")
                #simpan txt
260.                out.writerow(simpan)
                #simpan csv
261.
262.                a = sorted(simpan, key=float, revers
e=True)
263.                print "HASIL SORTING PROBABILITAS\n"
, a, "\n"
264.                simpan_file.write("HASIL SORTING PRO
BABILITAS: "+str(a)+"\n")
                #simpan txt
265.
266.                seq = a[:1]
267.                final_seq = ''.join(seq)
268.                print "Probabilitas tertinggi adalah
:", final_seq
269.                simpan_file.write("PROBABILITAS TERT
INGGI: "+str(final_seq)+"\n")
                #simpan txt
270.
271.                tanda=0
272.
273.                for num in range(0, j):
274.                    if final_seq==simpan[num]:
275.                        tanda += 1
276.
277.                    if tanda<=1:
278.
279.                        print "INI KETEMU"
280.                        flag=num
281.                        print "INI FLAG", flag

```

```

282.                                seq_labelset = labelset[
    num:]
283.                                seq_labelset_2 = seq_lab
    elset[:1]
284.                                final_seq_labelset = ''.
    join(seq_labelset_2)
285.                                print "TOPIK FINAL:", fi
    nal_seq_labelset
286.                                simpan_file.write("TOPIK
    FINAL DOKUMEN "+str(p)+": "+str(final_seq_labelset
    )+"\n\n")    #simpan txt
287.                                simpan_file_hasil_theta.
    write(str(final_seq_labelset)+"\n")
288.
289.                                print "TANDA: ADA", tanda, "ARRAY TE
    RATAS DENGAN NILAI YANG SAMA"
290.                                print '\n'
291.
292.                                print "TOTAL DOKUMEN:", d
293.
294.                                header_perplexity = '\n\n\n=====
    ===== PERPLEXITY =====
    ====='
295.                                print header_perplexity
296.                                perplexity = llda.perplexity()
297.
298.                                simpan_file.close()
299.                                simpan_file_hasil_theta.close()
300.                                print("SUKSES SIMPAN!")
301.
302.                                if __name__ == "__main__":
303.                                    start_time = time.time()
304.
305.                                    main()
306.
307.                                    time_akhir = time.time() - start_time
308.                                    sys.stderr.write("---
    "+str(time_akhir)+" detik ---")

```

Kode Sumber 28 Labeled Latent Dirichlet Allocation

```

1. import time, sys, time
2.
3. def send_to_database(skor_universitas, bobot_total,
    jumlah_dokumen, nama_universitas):
4.
5.     print skor_universitas
6.     print bobot_total
7.     print jumlah_dokumen
8.     print nama_universitas
9.
10.    nama_univ = nama_universitas
11.    print nama_univ
12.
13.    nilai_bobot_total = skor_universitas
14.    print nilai_bobot_total
15.
16.    tanggal = time.strftime('%Y-%m-%d %H:%M:%S')
17.    print tanggal
18.
19. def main():
20.     nama_file = "result_theta_all_Institut Pertania
    n Bogor.txt"
21.     fname = "D:/TA/Helmholtz/Code/Ger/llda/-
    tanpa pos tagging/result_llda_with_tf_idf_sklearn
    - helmholtz - all2017/"+nama_file
22.
23.     lala = fname[100:]
24.     nama_universitas = lala[:-4]
25.     print nama_universitas
26.
27.     a=""
28.     bobot=0
29.     bobot_total=0
30.     float(bobot_total)
31.
32.     with open(fname) as f:
33.         content = f.readlines()
34.
35.         content = [x.strip() for x in content]
36.
37.         #===== rule untuk pembobotan (refer
    ensi pak riyan) =====
38.         simpan_label = {}

```



```
39.     simpan_probs = {}
40.     jumlah_label=0
41.
42.     for i in range(0, len(content)):
43.         simpan_label[i] = content[i][:2]
44.         simpan_probs[i] = content[i][3:]
45.
46.         if(simpan_label[i]=="T1"):
47.             bobot = 10 * float(simpan_probs[i])
48.             bobot_total += bobot
49.
50.         elif(simpan_label[i]=="T2"):
51.             bobot = 20 * float(simpan_probs[i])
52.             bobot_total += bobot
53.
54.         elif(simpan_label[i]=="T3"):
55.             bobot = 30 * float(simpan_probs[i])
56.             bobot_total += bobot
57.
58.         elif(simpan_label[i]=="T4"):
59.             bobot = 40 * float(simpan_probs[i])
60.             bobot_total += bobot
61.
62.         elif(simpan_label[i]=="T5"):
63.             bobot = 50 * float(simpan_probs[i])
64.             bobot_total += bobot
65.
66.         elif(simpan_label[i]=="T6"):
67.             bobot = 60 * float(simpan_probs[i])
68.             bobot_total += bobot
69.
70.         elif(simpan_label[i]=="T7"):
71.             bobot = 70 * float(simpan_probs[i])
72.             bobot_total += bobot
73.
74.         elif(simpan_label[i]=="T8"):
75.             bobot = 80 * float(simpan_probs[i])
76.             bobot_total += bobot
77.
78.         elif(simpan_label[i]=="T9"):
79.             bobot = 90 * float(simpan_probs[i])
80.             bobot_total += bobot
81.
```

```

82.         jumlah_label+=1
83.
84.     jumlah_dokumen=9
85.     skor_universitas = float(bobot_total)/jumlah_dokumen
86.
87.     sys.stderr.write("JUMLAH LABEL TERDETEKSI: "+str(jumlah_label)+"\n")
88.     sys.stderr.write("BOBOT TOTAL: "+str(skor_universitas)+"\n")
89.
90. if __name__ == "__main__":
91.     start_time = time.time()
92.
93.     main()
94.
95.     time_akhir = time.time() - start_time
96.     sys.stderr.write("---
"+str(time_akhir)+" detik ---")

```

Kode Sumber 29 Pembobotan Dokumen

BIODATA PENULIS



Anugerah Yulindra Satyaji atau yang biasa disapa dengan nama Gerry lahir di Jakarta pada tanggal 11 Juli 1995. Memiliki seorang kakak laki-laki dan telah menempuh pendidikan di TK Sampaghita, SD Negeri Rawa Barat 05 Pagi (2001-2007), SMP Negeri 13 Jakarta (2007-2010), SMA Negeri 55 Jakarta (2010-2013) dan saat ini sedang menempuh pendidikan Sarjana di Teknik Informatika, Fakultas Teknologi Informasi, Institut Teknologi Sepuluh Nopember Surabaya angkatan tahun 2013.

Memiliki hobi menonton film dan tertarik dengan dunia tersebut. Terlibat aktif dalam organisasi kemahasiswaan serta kepanitiaan selama perkuliahan, antara lain staff Dalam Negeri di Himpunan Mahasiswa Teknik Computer-Informatika ITS 2014/2015, Staf NST (National Seminar and Technology) Schematics 2014, Staf Keamanan dan Perizinan acara Schematics 2015. Untuk komunikasi, dapat dihubungi melalui surel: gerrygerray@gmail.com